Trade-offs between reproductive and somatic investment in male birds

Andrew John Ferguson

This thesis is submitted for the degree of Doctor of Philosophy,
Division of Environmental and Evolutionary Biology,
Institute of Biomedical and Life Sciences,
University of Glasgow,
September, 2000.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Declaration

The work described in this thesis is my own unless otherwise stated. It is entirely of my own composition and has not, in whole or in part, been submitted for any other degree.

Andrew John Ferguson,
September, 2000

To My Mum and Dad

Acknowledgments

I am grateful to my supervisors, Dr Neil Metcalfe and Dr John Swaddle, for all their support and guidance, and to Dr Mark Witter for the part he played in the conception of this project. Special thanks go to Neil for neither leaving Glasgow nor resigning during my studentship.

I am also grateful to Stuart Humphries and the Ornithology Group for helpful discussions and ideas.

For financial support I thank my parents and the Natural Environment Research Council.

Abstract

Strategic allocation decisions, in particular trade-offs between reproductive and somatic investment, are central to life history theory. The allocation of resources to body mass maintenance is beneficial for avoiding starvation, but is costly because resources are diverted away from the current reproductive effort, and mass-dependent costs are increased. Stochastic dynamic models have been developed to investigate optimal routines of mass regulation and behaviour in male birds displaying to attract a mate. What is central to these models, and to any empirical investigation of the trade-off, is the dynamic interaction of body mass regulation and display behaviour routines. Previous experimental work has focussed on each of these factors in isolation, but never in unison. This work addressed this shortfall by examining the dynamic interaction of investment in reproduction versus somatic investment in male zebra finches (Taeniopygia guttata), a theme discussed more fully in chapter 1.

Chapter 2 provides details of software and hardware that I created in order to enable body mass and display hopping activity of males to be automatically recorded. Further, it describes software and hardware that controlled the lighting and temperature of the experimental environment and finally it discusses software I created to perform post-hoc processing of the data which were generated. The lighting controller sought to mimic the variations in light intensity associated with twilight. In chapter 3, I describe an experiment which tested the effect of naturalistic light intensity control versus 'all-or-nothing' lighting on the mass regulation and display behaviour of male zebra finches.

In the experiment described in chapter 4, I presented male zebra finches with four stimulus treatments which inherently varied in their potential to be courted, and I recorded diel patterns of body mass regulation and hopping activity. The stimulus cage contained either: no stimulus bird, a female bengalese finch, a male zebra finch, or a female zebra finch. There were significant effects of treatment on mean body masses and display hopping activity, with lowest mean masses and highest display activity during the female zebra finch treatment. Mass trajectories did not differ between treatments, instead the effect of treatment was manifested as a complete shifting in the y-axis of the mass curves. Hopping activity trajectories did however vary between

treatments with the female zebra finch stimulus evoking a marked peak around dawn. This apparent anomaly may be attributed to either strategic compensatory foraging aimed at guarding mass trajectory, variation in the mass-dependent costs of display behaviour, a reorganisation of time and energy budgets, or some combination of these factors. This work provided the first experimental evidence of the trade-off between reproductive and somatic investment in birds.

The diel patterns of display activity exhibited by many male passerines could be due to temporal variation in either male-state or in female-state (i.e. her attractiveness and/or receptivity). These effects are naturally confounded, but by desynchronising the photoperiods of the female stimulus birds from the males it was possible to disentangle the effects of the time of presentation and time-state of the female on male mass regulation and display activity. Female zebra finches were presented to the males for a discrete three hour period at dawn or midday, and in each case females were either in a dawn or a midday time-state (making four treatments in total). The time of presentation had no effect on overall display rates, but the peaks of display activity were shifted to coincide with the time when the female was present. Despite no difference in mean hopping activity, there was a significant effect of presentation time on mean mass, which I attributed to time of day differences, due to male-state, in the costs of display. Female time-state was for the first time shown to influence display activity of males. Dawn-state females were more attractive/responsive and evoked a greater display effort, but only in the context of the males' dawn, as no effects of female time-state were apparent at midday.

Arbitrary symmetry has been shown to influence female choice in birds, particularly in zebra finches. We investigated the effects of arbitrary symmetrical and asymmetrical traits of females (applied as coloured leg bands) on the dynamic interaction between body mass regulation and display activity in male zebra finches (chapter 6). We found no significant effects of leg band treatment on either overall levels of body mass and display hopping, or on their diel trajectories, all of which were highly consistent between treatments. There was however a significant and consistent effect of female identity (irrespective of leg banding) on male display hopping activity. This fact, coupled with evidence from chapters 4 & 5 that the trade-off exists and can be detected by the present experimental design, points towards arbitrary symmetrical traits of

female zebra finches having no effect (or a very weak effect) on male choice, unlike the situation that occurs when the sex roles are reversed.

Foraging is incompatible with courtship, therefore the decision to engage in courtship creates a conflict with mass regulation behaviour. Stochastic dynamic models predict that variability in patterns of food availability will influence the trade-off between investment in reproduction versus body mass maintenance, specifically that variable foraging success will cause mass to increase and display output to decrease. I examined the effects of food restrictions on mass regulation and display hopping activity of males that were presented with a female stimulus bird (chapter 7). The restriction treatments, which were not variable within a treatment, had significant effects on body mass trajectories, but had no significant effects on either mean body mass, directed hopping activity or the trajectory of hopping activity, despite their lasting for one third of the feeding daylength. There was in fact a non-significant tendency for hopping activity to be higher or highest during the period when food was restricted. A cost of courtship is time lost from foraging. Therefore when food is unavailable courtship activity becomes less costly but the risk of starvation is greater. Males appeared to overlook this increased risk and defended their display output.

The final chapter synthesises the findings of the previous chapters and discusses some ideas for further work in this area.

Contents

List of Tables

List of Figures

Chapter 1. General Introduction

Life History Trade-offs

The myriad strategic decisions an individual makes throughout its lifetime are the corner stone of life history theory. Whenever an individual makes a decision to allocate time or resources there is inevitably a trade-off involved. These trade-offs are paid for in terms of fitness, and costs arise whenever a beneficial change in one trait necessitates a detrimental change in another.

The most important life history trade-off relates to the costs of reproduction (Partridge & Harvey, 1985; Stearns, 1989). The costs of reproduction can themselves can be broken down into two principal components: the cost of reduced survival prospects and the cost of reduced future reproductive potential. Measurement of these trade-offs depends on the nature of the organism. Stearns (1989) defines two categories of breeding type as 'income breeders' and 'capital breeders', which in fact represent the two end-points of a continuum. Capital breeders, for example red deer (Cervus elaphus), store food as body fat that can be later utilised to fuel more than one reproduction attempt (Clutton-Brock et al., 1983). Consequently, the future reproductive success of capital breeders is inextricably linked to past success. For income breeders on the other hand, the resource pool is too small conceivably to pay for successive reproduction attempts and those species divert current income directly towards investment in reproduction. An excellent example of an income breeder is the ciliate protozoan Tokophrya lemnarum that feeds on Paramecium and produces one offspring for each food item it consumes (Kent, 1981). Great tits (Parus major) display some characteristics of both capital and income breeders. While some studies found no evidence of a trade-off between clutch size and adult survival (e.g. Pettifor et al., 1988) others found a trade-off between clutch size and success of a second brood (Smith et al., 1987; Tinbergen, 1987; Lindén, 1988). In this case the short-term costs within a season are clear whereas the longer-term costs are not.

A small opportunist breeder such as the zebra finch (Taeniopygia guttata) is likely to be a clearly defined income breeder. The Australian zebra finch, weighing typically 12 to

15 grams, lives naturally in an arid climate with a distribution covering the majority of that continent (Zann, 1996). They are renowned for their ability to breed at any time of year in response to rainfall that brings with it flushes of their staple food, grass seed (Morton & Davies, 1983; Zann et al., 1995). The absence of any distinct breeding period, and their reliance on rainfall to promote their food supply, limits the prospect of any element of capital breeding in their life history cycle, although there is evidence that female zebra finches deplete skeletal muscle protein during egg production (Houston et al., 1995a, 1995b).

While the costs of reproduction in a capital breeder will involve many elements of the organism's previous behaviour, the costs of reproduction in an income breeder can more easily be related to current behaviours, for example the diel pattern of foraging behaviour. It is most often the case, with some exceptions (e.g. courtship feeding seen in certain arthropods), that foraging and behaviours related to reproduction are mutually exclusive. This is certainly the case for the zebra finch (Zann et al., 1995; Zann, 1996). This fact, and the direct relationship between the costs of reproduction and current foraging behaviour, crystallises the trade-off between reproductive investment versus somatic investment in this species. This trade-off will be steered by short term dynamic allocation decisions, for example whether to increase fat reserves or not. Increasing fat reserves or body mass is beneficial as a buffer against starvation (e.g. McNamara et al., 1997) but is costly in terms of increasing mass-dependent costs (e.g. flight costs, Pennycuick, 1990; predation risk, Witter et al., 1994; Gosler et al., 1995; Metcalfe & Ure, 1995; risk of injury, Witter & Cuthill, 1993) and diverting energy away from the current reproductive effort (Witter & Cuthill, 1993). There is some correlational evidence of this trade-off between body mass reserves and the amount of mate solicitation behaviour in birds (black grouse, Tetrao tetrix, Rintamaki et al., 1995) and also in mammals (Richardson's ground squirrels, Spermophilus richardsonii, Michener 1998; red foxes Vulpes vulpes, Cavallini, 1998).

Diel Patterns of Display and the Dawn Chorus in Birds

One of the contexts in which the trade-off between reproductive and somatic investment has been most intensively studied is the courtship behaviour of male song birds. Many passerines illustrate the phenomenon of pronounced peaks in courtship song at specific times of day (the dawn and dusk 'choruses'). Most temperate zone passerines spend a higher proportion of their time singing at dawn than during the rest of the day, with a secondary peak at dusk (Kacelnik & Krebs, 1982) and this phenomenon has attracted much research into its cause and functions (e.g. Garson & Hunter, 1970; Mace, 1986, 1987a, 1987b, 1987c, , 1989a, 1989b, 1989c; McNamara et al., 1987; Cuthill & MacDonald, 1990; Staicer, 1996; Staicer et al., 1996; Strain & Mumme, 1988; Thomas, 1999a, 1999b).

There is no absolute consensus as to why birds opt to display more at dawn and different studies frequently suggest different functions or motivations. An important but frequently overlooked factor is the pairing status of the singing males (Mace, 1987b). A male who is seeking to attract a female and a male who already has a mate and seeks to guard her and protect his paternity have clearly different priorities, and this will likely be reflected in their display strategies. Displaying is an energetically expensive activity, and bears a further cost in terms of time lost from foraging, therefore one might intuitively expect it to occur more when it is least costly.

Several functional and causal hypotheses for the diel pattern of display behaviour of many passerines have been proposed involving environmental, energetic, social and sexual factors. Henwood & Fabrick (1979) developed a numerical model of sound propagation in the varying atmospheric conditions that arise at different times of day and concluded that early morning microclimatic conditions will strongly favour sound propagation, a conclusion also reached empirically (Waser & Waser, 1978). Whether an individual should seek to maximise the range over which his song is broadcast is unclear and depends on the functional reason behind the song (e.g. Mace, 1987b). What is clear is that the extent of the advantage that Henwood & Fabrick (1979) proposed will be diminished by interference by other birds' songs (e.g. Bremond, 1978), a factor that was not accounted for in their model.

Kacelnik (1979) demonstrated that foraging success was reduced by low light intensities and he argued that twilight was therefore an inefficient time for foraging, which subsequently promoted alternative behaviours such as singing or displaying. Due to the fact that foraging and displaying are incompatible with one another, a *de facto* cost of displaying is that of time lost from foraging (Kacelnik & Krebs, 1982). Mace (1987b) did not entirely agree with this statement and pointed out that foraging and singing do not always compete for space in an animal's time budget. Mace (1987b) went on to cite personal observations that indicated that great tits had free hours in their days. There is little question that many bird species do spend a substantial portion of their day being inactive (e.g. European starlings (Sturnus vulgaris), Bautista et al., 1998; zebra finches, Dall & Witter, 1998), but since the pattern of inactivity varies with time, this does not imply that the extent to which foraging and singing compete for time may not vary with specific times of day (e.g. at dawn).

The dawn chorus may arise because early morning circumstances favour territorial incursions (Kacelnik & Krebs, 1982) or because of mate guarding considerations linked to diel patterns of female fertility (Mace, 1987a; Møller, 1991a). Territorial challenges may be concentrated in the early hours of the morning due to an accumulation of vacant territories due to overnight mortalities, or an accumulation of migrants arriving overnight (Kacelnik & Krebs, 1982). If display behaviour has a territorial function one might therefore expect it to be more concentrated at dawn. Mace (1987a) demonstrated that the song output of male great tits was directly related to female fertility and the female's position in the egg-laying sequence. Most passerines lay their eggs at dawn and evidence from other species suggests that fertility peaks immediately afterwards (Sturkie, 1965; Howarth, 1971; Cheng et al., 1983) which would in turn favour proximity to, and mate-guarding of the female early in the day, an effect already demonstrated (Mace, 1989b). It has also been demonstrated, again in great tits, that the dawn chorus is influenced by interactions between the female and male. Mace (1986) found a strong correlation between male song duration and the time of emergence of the female from her nest hole. Possible functions for this mechanism may include stimulating the female to emerge and copulate and/or a warning signal to other males to deter intrusions and/or extra-pair copulations.

The idea of singing occurring when it is least costly/most beneficial was contradicted by the suggestion that choruses occurred at dawn and dusk precisely because these were in fact the costliest times to sing, and therefore constituted particularly good times for male quality to be assessed by females (Montgomerie, 1985, as cited by Cuthill & MacDonald, 1990). To be an honest signal of male quality, display behaviour has to be costly (Mace 1987a, 1987c; Grafen, 1990; Hutchinson et al., 1993; Otter et al., 1997) as it clearly is, but both the costs of display and of competing activities are variable with time of day (McNamara et al., 1987). Food supplementation experiments (Gottlander, 1987; Strain & Mumme, 1988; Cuthill & MacDonald, 1990; Lucas, 1999; Thomas, 1999a, 1999b) have shown that supplementation increased song output. This has been interpreted as suggesting that song constituted an honest advertisement (Cuthill & MacDonald, 1990). However, increased song output in these experiments was merely a reflection of an artificial short-term enhancement in energetic status. Given the manner in which experimental supplementations were delivered, access to or utilisation of those supplementations would vary little with direct measures of quality such as territory quality, parasite burden, phenotype or genotype. To determine, therefore, if song was indeed an honest advertisement of male quality it would be more appropriate to manipulate quality directly and measure its effect on song output with time of day. For example, it has recently been shown that male Beaugregory damselfish (Stegastes leucosticus) displayed more vigorously when the quality of their territory was enhanced (Itzkowitz & Haley, 1999), and that, in sedge warblers (Acrocephalus schoenobaenus), there is a negative correlation between blood-borne parasite burden and both song output and repertoire (Buchanan et al., 1999).

Life History trade-offs: predictions of theoretical models

The specific trade-off between reproductive and somatic investment, and the dynamic allocation decisions that bear an influence upon it, have attracted the attention of theoreticians in recent years (Houston & McNamara, 1987; McNamara & Houston, 1987; Hutchinson et al., 1993). Their stochastic dynamic programming models looked specifically at the cost of male singing behaviour as a readily quantifiable part of the total costs of reproduction, but are relevant to any display behaviour as long as it is both costly and incompatible with foraging (Hutchinson et al., 1993).

Houston & McNamara (1987) described how a dynamic optimisation model, in which an animal makes a series of decisions that both vary in payoff and affect its state, can be converted into a stochastic dynamic game that can subsequently be solved using computational methods. Their technique finds the evolutionary stable strategy (ESS) for an individual interacting with other individuals within a population. It was the programming techniques described in this paper that were employed by McNamara et al. (1987) in their model of the optimal organisation of the daily routines of singing and foraging in small birds. This model considered how an unpaired male bird should divide his time between two mutually exclusive behaviours; singing or foraging. In general there is a critical boundary of body reserves above which it is optimal to sing to obtain and/or defend a mate or territory and below which it is optimal to forage, and the level of this boundary varies with time of day. The model assumed variable food rewards of foraging and accounted for the mass-dependent costs of singing and foraging. Using these rules the authors were able to calculate optimal singing and foraging strategies.

With many of the parameters of their model, the effects of changing the mean and altering the variance had separate and qualitatively different effects on the optimal behaviour pattern, a point that must be borne in mind when empirically testing any of the predictions of the model (see Cuthill et al., 2000). An example of such a parameter was the overnight energy requirement. Increasing the mean level of overnight expenditure required birds to attain a higher level of reserves by dusk (when feeding ceased), which simply meant that the optimal strategy was to start building reserves earlier in the day. However, body reserves at dawn and consequently dawn song output were unaffected. In contrast, while changing the variance of overnight requirements had the same effect of increasing reserves at dusk, it meant there was variability in dawn state with a majority of individuals arriving at dawn with reserves above the critical boundary. This brought about a dawn chorus with a longer duration, and the greater the overnight variance the longer the dawn chorus. One factor that is predicted to increase overnight expenditure is ambient temperature. It has been shown empirically that there is a positive correlation between overnight temperature and dawn song output in wrens (Troglodytes troglodytes; Garson & Hunter, 1979), Carolina wrens (Thryothorus ludovicianus; Strain & Mumme, 1988) and in pied flycatchers (Ficedula hypoleuca; Gottlander, 1987).

McNamara et al. (1987) examined the effect of introducing a probability of foraging being interrupted into the model. Such stochasticity of food supply led to individuals maintaining a higher level of reserves throughout the day. In that situation birds began building their reserves earlier in the day and hence reached their target dusk mass more rapidly, which increased dusk song output but had no effect on dawn output.

McNamara et al. (1987) also explored temporal changes in the profitability of both singing and foraging and the effects of these changes on the optimal strategy. As discussed earlier, several factors have been proposed that may make song more profitable at dawn, for example it may be favoured by atmospheric conditions (Henwood & Fabrick, 1979) or the probability of attracting a mate may be greater (Kacelnik & Krebs, 1982; Mace, 1986). Increasing the profitability of song at dawn in the model led to a more intense dawn chorus but less dusk output as males sought to build their reserves earlier to place them in a more favourable position at the start of the next day. It has also been argued that foraging may be less profitable at dawn due to low light levels (Kacelnik, 1979; Kacelnik & Krebs, 1982) or possibly low temperature (Avery & Krebs, 1984), although the latter applied only to birds foraging on prey such as arthropods that may be more torpid and therefore harder to find in colder conditions. The model of McNamara et al. (1987) indicated that if foraging profitability peaked around midday then most foraging should occur just after that peak and singing should be biased towards dawn.

The predictions of McNamara et al. (1987) were on the whole characteristic of the typical song output of small passerines and were robust to changes in a wide range of parameters. While previous work on the dawn chorus had focussed on diurnal changes in the profitability of singing (e.g. Mace, 1986, 1987a) or foraging (e.g. Kacelnik & Krebs, 1982) as functional explanations, their model suggested that diel patterns of singing could arise even when extrinsic/environmental considerations such as these were held constant. As such, their model was one of the first functional models capable of explaining diel patterns in behaviour.

Hutchinson et al. (1993) took up the theme of McNamara et al. (1987) and developed the idea further to include differences both in male quality and in female assessment of song (which was allowed to vary with either time of day or experience of previous song

output). This permitted them to determine the optimal singing routine for each male under each female's decision rule and also to determine the efficiency of the female rule in assessing variable male quality. Singing is incompatible with foraging and it also increases the risk of starvation, therefore singing functions as an honest signal of male quality (e.g. Mace, 1987b; Staicer, 1996, Staicer et al., 1996; Otter et al., 1997). The model of Hutchinson et al. (1993) allowed an insight into this role of song as a strategic handicap conveying honest information about male quality or territory quality (Grafen, 1990).

Courtship in zebra finches

In the present work we treated the stereotypical parallel hopping activity that male zebra finches engage in during courtship as analogous to the mate attraction function of song in songbirds. There are two stages of male courtship that precede copulation in zebra finches (Morris, 1954; Immelman 1959, 1962 as cited by Zann, 1996; Zann, 1996). The majority of the courtship sequence is composed of stage 1 behaviours where the male firstly flies to and fro between perches, before making a series of short hops between adjacent perches (parallel hopping). These behaviours are typically performed first by the male and then are usually reciprocated by the female. The second stage involves similar parallel hopping behaviour that may be accompanied by vocalisations on the part of the male. The female may again reciprocate the hopping behaviour and there is a significant tendency for the male to favour the presentation of his flanks, particularly his right flank (Workman & Andrew, 1986; Zann, 1996).

This form of display behaviour meets the criteria required by the previously discussed theoretical models (Houston & McNamara, 1987, McNamara et al., 1987; Hutchinson et al., 1993) as it is clearly both energetically expensive and incompatible with foraging activity. The models made the assumption that the costs of singing were mass-dependent. The consequence of this was that it was predicted to be less costly for the male to sing early in the day (before his mass increased as a result of foraging), which in turn biased early morning behaviour towards singing (and thus gave rise to a dawn chorus in the absence of any circadian patterns in the environment). The degree to which the energetic costs of singing are mass-dependent have never been tested empirically. However, given that singing is a stationary behaviour, involving

musculature including that associated with the beak and pulmonary function, it is very difficult to imagine any scope for significant mass-dependence. At a crude level it is immediately apparent that attaching weights to a singing bird does not increase the energetic cost of singing (c.f. Witter et al., 1994). However, the costs of the parallel hopping display of male zebra finches, being a predominantly locomotory behaviour involving a high frequency of starting and stopping, will clearly be strongly mass-dependent. McNamara et al. (1987) found that introducing even small mass-dependent costs of displaying into their model generated distinct policies of daily behaviour. The predictions of such models may therefore be more relevant to relatively strongly mass-dependent display behaviours such as that of male zebra finches, compared to singing where the level of mass-dependence is at best very small.

The display behaviour of male zebra finches lends itself very well to automatic data collection techniques. Microswitch perch mechanisms, which recorded the temporal and spatial pattern of hopping, were first employed by Swaddle & Cuthill (1994a, 1994b), and design details of those used in the present study are given in Chapter 2. Employing an automatic data collection mechanism ensured that the complete diurnal activity (and hence courtship) record for each experimental subject was recorded, something that is practically impossible for studies involving birdsong where subjects' behaviours are often scanned alternately (e.g. Gottlander, 1987; Mace, 1989a; Cuthill et al, 1990; Thomas, 1999a, 1999b).

Zebra finches have been the species of choice for much of the laboratory based research into avian biology. Although they adapt well to captivity, the species has been captive bred in Europe for over a century and domesticated individuals are the principal source of stock for researchers. Zebra finches are a highly social species which form flocks in the wild of up to several hundred individuals. A consequence of this sociality is that birds can be housed in the laboratory in groups (typically of no more than eight individuals) which permits economic use of space. Their small size and wingspan allows birds to exercise their flight muscles even in relatively small cages, and the absence of outside flight space has not been shown to be of any particular detriment to their welfare. The husbandry of captive zebra finches is very straightforward. In nature birds consume a diet almost exclusively of small grass seeds although insectivory has infrequently been reported (Immelmann, 1962; Immelmann, 1970; as cited by Zann,

1996). In captivity zebra finches thrive on proprietary grass seed mixes (e.g. Haiths' Foreign Finch Mix, containing Panicum millet, yellow millet, Japanese millet, pearl white millet, red millet and Canadian canary seed: Haiths, Cleethorpes, U.K.) and also benefit from the provision of cuttlebone and shell-grit for the maintenance of their beak and digestion respectively.

As discussed already, zebra finches are opportunistic breeders provided that nutritional state permits, and this willingness to breed throughout the year makes the species ideal for research involving any aspect of sexual selection or reproductive behaviour, as well as facilitating the breeding of additional stock. Zebra finches are socially monogamous and generally form pair bonds which last for life, however, if a pair bond is disrupted by death of a partner or by experimental manipulation, birds will re-pair if provided with the opportunity (Zann, 1996). Consequently, a male will readily court a new female if separated from a previous mate and in fact a male will readily court several females in succession if given the opportunity (e.g. Silcox & Evans, 1982). This fact has obvious implications for sexual selection studies involving a choice of stimuli whether presented simultaneously or serially.

Forming Hypotheses: factors affecting the trade-off between investment in soma versus reproduction

Many empirical studies have tested some of the modelling predictions discussed above, particularly those concerned with the effect of energetic status on song routines (e.g. Gottlander, 1987; Strain & Mumme, 1988; Cuthill & MacDonald, 1990; Lucas, 1999; Thomas, 1999a, 1999b). In the laboratory, energetic status has been manipulated by adjustments in ambient temperature (e.g. Ekman & Hake, 1990; Bednekoff et al., 1994; Lilliendahl et al., 1996; Meijer et al., 1996), wind chill (Witter et al., 1994; Cuthill et al., 2000) and food availability (e.g. Mace, 1989b; Bednekoff & Krebs, 1995). In the field, the putative effects of ambient temperature/weather on energetic status (e.g. Gottlander, 1987; Strain & Mumme, 1988) along with the effects of food supplementations on song output (e.g. Cuthill & MacDonald, 1990) have also been measured. Manipulations of food availability have taken the form of restrictions (e.g. Kacelnik & Krebs, 1982; Dall & Witter, 1998) and/or supplementations (e.g. Cuthill & MacDonald, 1990; Thomas

1999a, 1999b) but frequently the effects of predictability and variability have been confounded (see Cuthill et al., 2000 for a discussion).

The central theme of the models discussed above however, is the dynamic interaction between body mass regulation and investment in reproductive behaviour, and how this is affected by short-term strategic allocation decisions. This trade-off, and the proximate factors that influence it, has never before been studied. The present work addressed this shortfall by considering the influence of certain factors on the dynamic interaction of investment in soma versus investment in reproductive activities in male zebra finches.

The work described in chapter 4, Trade-off between body mass and reproductive behaviour in male zebra finches, examined the effect of the opportunity to attract a mate on the diurnal strategies of display hopping and body mass regulation of males. When a male was presented with a potential mate we predicted that he should invest more time and energy in courtship activity, which may detract from the amount of time and energy he devoted to feeding, hence altering his mean body mass and his diel pattern of body mass change (trajectory). I presented the male with four stimulus treatments, one of which was a female zebra finch, the other three being controls to allow me to distinguish between the effects of a female conspecific as opposed to the effects due simply to the presence of another bird.

Amongst other possible factors, the typical pattern of male display in many passerines could be due to temporal variation in either male state, or in the receptivity of the female (c.f. Mace, 1987a; Møller, 1991a). In chapter 5, The effects of time of day and context of female presentation on body mass and display in male zebra finches, I explored these effects by presenting males with stimulus females for discrete periods at dawn and midday. The natural association between the time-states of the two birds was removed by desynchronising the females relative to the males' photoperiod, such that females were themselves in either a dawn or a midday time-state when presented to the males. A common assumption of many models of mate choice is that males are equally receptive to all females (for a review see Reid & Stamps, 1997). However this has frequently been shown not to be the case (e.g. Burley, 1981; Burley et al., 1982; Verrel, 1982; Olsson, 1983; Sargent et al., 1986; Monaghan et al., 1996; Amundsen et al., 1997; Hansen et al., 1999; Itzkowitz & Haley, 1999). I therefore tested the hypothesis that

both the time of presentation and the time-state of a female would affect the dynamic interaction of display hopping and body mass regulation in male zebra finches.

Natural fluctuating asymmetry (Møller, 1990; Møller, 1991b; Møller, 1992a, 1992b; Manning & Chamberlain, 1993; Swaddle & Cuthill, 1995), manipulation of the asymmetry of natural traits (Møller, 1989; Oakes & Barnard, 1994; Swaddle & Cuthill, 1994a; Morris & Casey, 1998) and the application of arbitrary asymmetrical traits (Swaddle & Cuthill, 1994b; Fiske & Amundsen, 1997; Bennett et al., 1996; Hansen et al., 1999; Waas & Wordsworth, 1999) have all been shown to either directly influence sexual selection mechanisms or have been found to be correlated with other measures of sexual selection. The direction of the preferences are sometimes inconsistent between studies and between species, and recently the role of natural asymmetry as a viable biological signal has been cast into doubt (Swaddle, 1999; Swaddle, 2000). In birds however, the effect on female choice of applying coloured leg bands to males in symmetric or asymmetric arrangements is in general very consistent (Swaddle & Cuthill, 1994b; Bennett et al., 1996; Swaddle, 1996; Fiske & Amundsen, 1997; Waas & Wordsworth, 1999; but see Jennions, 1998 for an exception) and recently similar effects have been demonstrated on male choice of females (Hansen et al., 1999).

Although male mate choice is markedly less prevalent than female choice, it has been demonstrated in a number of species (e.g. Verrel, 1982; Olsson, 1983; Sargent et al., 1986; Amundsen et al., 1997; Hansen et al., 1999; Itzkowitz & Haley, 1999) including the zebra finch (Burley, 1981; Burley et al., 1982; Monaghan et al., 1996). Female mate choice is favoured by phenomena such as anisogamy and female-biased parental investment, however, some degree of male choice is expected in monogamous species (Andersson, 1994) or those in which breeding is highly synchronised (Owens & Thompson, 1994; Kuester & Paul, 1996; Amundsen et al., 1997), for example the zebra finch. In chapter 6, Arbitrary symmetrical traits on females: their effect on diel patterns of display and mass regulation in male zebra finches, I manipulated an arbitrary symmetrical trait (coloured plastic leg bands) of stimulus females with the intention of varying their attractiveness to males. I then assessed the effect of leg band manipulations on diel patterns of display hopping activity and body mass regulation to test the prediction that males would trade-off body mass maintenance behaviours in

favour of more vigorous display in front of putatively more attractive (symmetrically banded) females, and vice versa (c.f. Hansen et al., 1999).

Theoretical models make certain predictions relating to food supply and foraging success. For example, if the gain from foraging becomes less predictable due to a probability of being interrupted, individuals maintain higher masses throughout the day that results in higher dusk display output but no change in dawn display output (McNamara et al., 1987). Alternatively, if the energy available to an individual is reduced, for example by decreasing the hours of daylight, or decreasing the net gain from foraging, a shorter dawn chorus with a lower magnitude is predicted (Hutchinson et al., 1993). In chapter 7, The effects of food restriction on diel patterns of body mass and display activity in male zebra finches, I investigated the effect of reducing the length of the feeding day by imposing food restrictions, thereby increasing the probability of feeding being interrupted relative to the control treatment (where there was no interruption). The timing of the interruption was constant within a treatment and variable between treatments.

The predictions of Hutchinson et al. (1993) assumed that a reduced daylength also decreased the energy available to a bird which did not allow for the possibility of compensatory foraging when feeding conditions were particularly good, while the predictions of McNamara et al. (1987) envisaged the timing of interruptions being variable both within and between days. My experiment therefore did not explicitly test predictions of either model as birds enjoyed *ad libitum* food availability outwith restriction periods and the timing of my restrictions were constant within a treatment. By restricting food during distinct periods in an individual's day, I explored the short-term consequences for body mass maintenance and display hopping activity with particular reference to the period of restriction. I predicted that during the third of the day when food was unavailable, individuals would seek to limit their mass loss by reducing energetic activities such as display behaviour. However, the extent to which individuals varied the trade-off between body mass maintenance and display activity when faced with food restriction was predicted to vary also with the timing of the restriction. Display behaviour appears to be more important at certain times of day (e.g. dawn) therefore individuals were predicted to guard their performance of that

behaviour, at those times of day when it is typically performed most, with the decrease in body mass and increase in starvation risk it would entail.

Chapter 3, Sigmoidal versus rectangular lighting regimes: effects on diel patterns of body mass and display hopping in male zebra finches, explored the practicalities and possible benefits of using a lighting system in the laboratory that sought to replicate the gradual light intensity changes associated with a natural dawn and dusk. A tremendous number of studies are performed using caged birds, particularly the zebra finch (see Zann, 1996), and while some of these studies have used natural light to illuminate the laboratory (e.g. Meijer et al., 1996; Pravosudov & Grubb, 1997), the majority have used artificial light sources or have failed to specify their lighting conditions. Laboratory studies allow experimenters the opportunity to precisely control many aspects of their subjects' environment. In the field or in outdoor aviaries, in contrast, it may be impossible to control temperature, weather, ambient olfactory or auditory signals and provocation by wild or domestic animals, any of which may significantly affect an individual's state (e.g. Garson & Hunter, 1970; Bremond, 1978; Witter et al., 1994; Lilliendahl, 1997, 1998).

Artificial laboratory lights are traditionally switched on and off in unison, with light intensity changing instantaneously from zero to full and *vice versa*. This situation is clearly very unnatural and denies subjects of any extrinsic cues relating to the imminent start and finish of the feeding day, despite patterns of foraging and display in passerines being intimately related to changes in light intensity at twilight. I therefore created a system that attempted to mimic these changing light patterns. Given the particular contribution of dawn and dusk to the diel patterns of display activity and body mass regulation, I predicted that the nature of the lighting regime (naturalistic versus all-or-nothing) would influence the dynamic interaction of these two behaviours.

Control of the laboratory environment and data collection within this series of experiments were fully automated and therefore relied heavily on technology in terms both of hardware and software. Such an automated system was, I believe, not only a very attractive but also a very valuable means of performing behavioural ecology experiments such as these. The principal advantages of this system were the sheer quality, quantity and reliability of data generated, the minimising of disturbance to the

experimental subjects and the time savings made. Computer control of the lighting and ventilation in the laboratory permitted the naturalistic light intensity changes described in chapter 3, and also ensured that the temperature of the room was kept within acceptable tolerances. The combined electronic balance/microswitch perch units and the logger software generated precise records of the temporal patterns of body mass and display hopping activity. All of these systems are described more fully in chapter 2.

Chapters 3, 4, 5, 6 & 7 of this thesis have been written in the style of manuscripts for submission for publication in peer-reviewed journals, and as such are co-authored with my supervisors, John P. Swaddle and Neil B. Metcalfe.

References

Amundsen, T., Forsgren, E. & Hansen, L. T. T. 1997. On the function of female ornaments: male bluethroats prefer colourful females. Proceedings of the Royal Society of London Series B, 264, 1579-1586.

Andersson, M. 1994. Sexual Selection. Princeton University Press: Princeton.

Avery, M. I. & Krebs, J. R. 1984. Temperature and foraging success of Great Tits Parus major hunting for spiders. Ibis, 126, 33-38.

Bautista, L. M., Tinbergen, J., Wiersma, P. & Kacelnik, A. 1998. Optimal foraging and beyond: how starling cope with changes in food availability. American Naturalist, 152, 543-561.

Bednekoff, P. A. & Krebs, J. R. 1995. Great Tit fat reserves: effects of changing and unpredictable feeding day length. Functional Ecology, 9, 457-462.

Bednekoff, P.A., Biebach, H. & Krebs, J. 1994. Great tit fat reserves under unpredictable temperatures. Journal of Avian Biology, 25, 156-160.

Bennett, A. T. D., Cuthill, I. C., Partridge, J. C. & Maier, E. J. 1996. Ultraviolet vision and mate choice in zebra finches. Nature, 380, 433-434.

Bremond, J. C. 1978. Acoustic competition between the song of the wren (Troglodytes troglodytes) and the song of other species. Behaviour, 65, 89-98.

Buchanan, K. L., Catchpole, C. K., Lewis, J. W. & Lodge, A. 1999. Song as an indicator of parasitism in the sedge warbler. Animal Behaviour, 57, 307-314.

Burley, N. 1981. Sex Ratio Manipulation and Selection for Attractiveness. Science, 211, 721-722.

Burley, N., Krantzberg, G. & Radman, P. 1982. Influence of colour-banding on the conspecific preferences of zebra finches. Animal Behaviour, 30, 444-455.

Cavallini, P. 1998. Differential investment in mating by red foxes. Journal of Mammalogy, 79, 215-221.

Cheng, K. M., Burns, J. T. & McKinney, F. 1983. Forced copulation in captive Mallards: III. Sperm competition. The Auk, 100, 302-310.

Clutton-Brock, T. H., Guinness, F. E. & Albon, S. D. 1983. The cost of reproduction to red deer hinds. Journal of Animal Ecology, 52, 367-383.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Cuthill, I. C., Maddocks, S. A., Weall, C. V. & Jones, E. K. M. 2000. Body mass regulation in response to changes in feeding predictability and overnight energy expenditure. Behavioral Ecology, 11, 189-195.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Ekman, J. B. & Hake, M. K. 1990. Monitoring starvation risk: adjustments of body reserves in greenfinches (Carduelis chloris L.) during periods of unpredictable foraging success. Behavioral Ecology, 1, 62-67.

Fiske, P. & Amundsen, T. 1997. Female bluethroats prefer males with symmetric colour bands. Animal Behaviour, 54, 81-87.

Garson, P. J., Hunter, M. L. 1970. Effects of temperature and time of year on the singing behaviour of wrens Troglodytes troglodytes and great tits Parus major. Ibis, 121, 481-487.

Gosler, A. G., Greenwood, J. J. D. & Perrins, C. 1995. Predation risk and the cost of being fat. Nature, 377, 621-623.

Gottlander, K. 1987. Variation in the song rate of the male pied flycatcher Ficedula hypoleuca: causes and consequences. Animal Behaviour, 35, 1037-1043.

Grafen, A. 1990. Biological signals as handicaps. Journal of Theoretical Biology, 144, 517-546.

Hansen, L. T. T., Amundsen, T. & Forsgren, E. 1999. Symmetry: attractive not only to females. Proceedings of the Royal Society of London Series B, 266, 1235-1240.

Henwood, K. & Fabrick, A. 1979. A quantitative analysis of the dawn chorus: temporal selection for communicatory optimisation. American Naturalist, 114, 260-274.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Houston, D. C., Donnan, D. & Jones, P. J. 1995. The source of the nutrients required for egg production in zebra finches Poephila guttata. Journal of the Zoological Society of London, 235, 469-483.

Houston, D. C., Donnan, D., Jones, P., Hamilton, I. & Osborne, D. 1995. Changes in the muscle condition of female Zebra Finches Poephila guttata during egg laying and the role of protein storage in bird skeletal muscle. Ibis, 137, 322-328.

Howarth, B. 1971. Transport of spermatozoa in the reproductive tract of turkey hens. Poultry Science, 50, 84-84.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Immelmann, K. 1959. Exerimentelle untersuchungen über die biologische bedeutung artsspezifischer Merkmale beim Zebrafinken (Taeniopygia guttata). Zoologische

Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 86, 437-592.

Immelmann, K. 1962. Beiträge zu einer vergleichenden Biologie australische Prachtfinken (Spermestidae). Zoologische Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 90, 1-196.

Immelmann, K. 1970. Der Zebrafink (Taeniopygia guttata). A. Ziemsen, Wittenberg Lutherstadt.

Itzkowitz, M. & Haley, M. 1999. Are males with more attractive resources more selective in their mate preferences? A test in a polygynous species. Behavioral Ecology, 10, 366-371.

Jennions, M. D. 1998. The effect of leg band symmetry on female-male association in zebra finches. Animal Behaviour. 55, 61-67.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

Kacelnik, A. 1979. The foraging efficiency of great tits (Parus major) in relation to light intensity. Animal Behaviour, 27, 237-241.

Kent, E. B. 1981. Life-history responses to resource variation in a sessile predator, the ciliate protozoan Tokophyra lemnarum Stein, Ecology, 62, 296-302.

Kuester, J. & Paul, A. 1996. Female-female competition and male mate choice in Barbary macaques (Macaca sylvanus). Behaviour, 133, 763-790.

Lilliendahl, K. 1997. The effect of predator presence on body mass in captive greenfinches. Animal Behaviour, 53, 75-81.

Lilliendahl, K. 1998. Yellowhammers get fatter in the presence of a predator. Animal Behaviour, 55, 1335-1340.

Lilliendahl, K., Carlson, A., Welander, J. & Ekman, J. B. 1996. Behavioural control of daily fattening in great tits (Parus major). Canadian Journal of Zoology, 74, 1612-1616.

Lindén, M. 1988. Reproductive trade-off between first and second clutches in the great tit Parus major: an experimental study. Oikos, 51, 285-290.

Lucas, J. R., Schraeder, A. & Jackson, C. 1999. Carolina chickadees (Aves, Paridae, Poecile carolinensis) vocalization rates: effects of body mass and food availability under aviary conditions. Ethology, 105, 503-520.

Mace, R. 1986. Importance of female behaviour in the dawn chorus. Animal Behaviour, 34, 621-622.

Mace, R. 1987a. The dawn chorus in the great tit Parus major is directly related to female fertility. Nature, 330, 745-746.

Mace, R. 1987b. Why do birds sing at dawn? Ardea, 75, 123-132.

Mace, R. 1987c. The dawn chorus: behavioural organisation in the great tit (Parus major). Unpublished D Phil Thesis, University of Oxford.

Mace, R. 1989a. A comparison of great tits' (Parus major) use of time in different daylengths at three European sites. Journal of Animal Ecology, 58, 143-151.

Mace, R. 1989b. Great tits choose between food and proximity to a mate: the effect of time of day. Behavioural Ecology and Sociobiology, 24, 285-290.

Mace, R. 1989c. The relationship between daily routines of singing and foraging: an experiment on captive great tits (Parus major). Ibis, 131, 415-420.

Manning, L. T. & Chamberlain, A. T. 1993. Fluctuating asymmetry, sexual selection and canine teeth in primates. Proceedings of the Royal Society of London Series B, 251, 83-87.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Meijer, T., Rozman, J., Schulte, M. & Stach-Dreesmann, C. 1996. New findings in body mass regulation in zebra finches (Taeniopygia guttata) in response to photoperiod and temperature. Journal of Zoology, 240, 717-734.

Metcalfe, N. B. & Ure, S. E. 1995. Diurnal variation in flight performance and hence potential predation risk in small birds. Proceedings of the Royal Society of London Series B, 261, 395-400.

Michener, G.R. 1998. Sexual differences in reproductive effort of Richardson's ground squirrels. Journal of Mammalogy, 79, 1-19.

Møller, A. P. 1989. Viability costs of male tail ornaments in a swallow. Nature, 339, 132-135.

Møller, A. P. 1990, Fluctuating asymmetry in Male Sexual Ornaments May Reliably Reveal Male Quality. Animal Behaviour, 40, 1185-1187.

Møller, A. P. 1991a. Why mated songbirds sing so much: mate guarding and male announcement of mate fertility status. American Naturalist, 138, 994-1014.

Møller, A. P. 1991b. Sexual ornament size and the cost of fluctuating asymmetry. Proceedings of the Royal Society of London Series B, 243, 59-62.

Møller, A. P. 1992a. Female swallow preference for symmetrical male sexual ornaments. Nature, 357, 238-240.

Møller, A. P. 1992b. Patterns of fluctuating asymmetry in weapons: evidence for reliable signalling of quality in beetle horns and bird spurs. Proceedings of the Royal Society of London Series B, 248, 199-206.

Monaghan, P., Metcalfe, N. B. & Houston, D. C. 1996. Male finches selectively pair with fecund females. Proceedings of the Royal Society of London Series B, 263, 1183-1186.

Montgomerie, R. D. 1985. Why do birds sing at dawn? Abstracts of spoken papers, 19th International Ecology Congress, 1, 242.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Morton, S. R. & Davies, P. H. 1983. Food of the zebra finch (Poephila guttata) and an examination of granivory in birds of the Australian arid zone. Australian Journal of Ecology, 8, 235-243.

Oakes, E. J. & Barnard, P. 1994. Fluctuating asymmetry and mate choice in paradise whydahs, Vidua paradisaea: an experimental manipulation. Animal Behaviour, 48, 937-943.

Olsson, M. 1983. Male preference for large females and assortative mating for body size in the sand lizard (Lacerta agilis). Behavioural Ecology and Sociobiology, 31, 337-341.

Otter, K., Chruszcz, B. & Ratcliffe, L. 1997. Honest advertisement and song output during the dawn chorus of black-capped chickadees. Behavioral Ecology, 8, 167-173.

Owens, I. P. F. & Thompson, D. B. A. 1994. Sex differences, sex ratios and sex roles. Proceedings of the Royal Society of London Series B, 258, 93-99.

Partridge, L. & Harvey, P. 1985. Costs of reproduction. Nature, 316, 20.

Pennycuick, C. J. 1990. Bird flight performance. Oxford University Press: Oxford.

Pettifor, R. A., Perrins, C. M. & McCleery, R. H. 1988. Individual optimization of clutch size in great tits. Nature, 336, 160-162.

Pravosudov, V. V. & Grubb, T. C. 1998. Management of fat reserves in tufted titmice (Parus bicolor): evidence against a tradeoff with food hoards. Behavioural Ecology and Sociobiology, 42, 57-62.

Reid, M. L. & Stamps, J. A. 1997. Female mate choice tactics in a resource based mating system: field tests of alternative models. American Naturalist, 150, 98-121.

Rintamaki, P. T., Alatalo, R. V., Höglund, J. and Lundberg, A. 1995. Mate sampling behavior of black grouse females (Tetrao tetrix). Behavioural Ecology and Sociobiology, 37, 209-215.

Sargent, R. C., Gross, M. R. & Van Den Berghe, E. P. 1986. Male mate choice in fishes. Animal Behaviour, 34, 545-550.

Silcox, A. P. & Evans, S. M. 1982. Factors affecting the formation and maintenance of pair bonds in the zebra finch, Taeniopygia guttata. Animal Behaviour, 30, 1237-1243.

Smith, H. G., Källander, H. & Nilsson, G. A. 1987. Effects of experimentally altered brood size on frequency and timing of second clutches in the great tit. The Auk, 104, 700-706.

Staicer, C. A. 1996. Honest advertisement of pairing status: evidence from a tropical resident wood-warbler. Animal Behaviour, 51, 375-390

Staicer, C. A., Spector D. A. & Horn, A. G. 1996. The dawn chorus and other diel patterns in acoustic signalling. In: Ecology and Evolution of Acoustic

Communication in Birds (Editors D.E. Kroodsma and E. H. Miller), pps. 426-453. Ithaca, New York: Cornell University Press.

Stearns, S. C. 1989. Trade-offs in life-history evolution. Functional Ecology, 3, 259-268.

Strain, J. G. & Mumme, R. L. 1988. Effects of food supplementation, song playback, and temperature on vocal territorial behavior of Carolina wrens. The Auk, 105, 11-16.

Sturkie, P. D. Avian Physiology, 2nd Edition. Balliere, Tindall & Cassell: London.

Swaddle, J. P. & Cuthill, I. C. 1994a. Female zebra finches prefer males with symmetric chest plumage. Proceedings of the Royal Society of London Series B, 258, 267-271.

Swaddle, J. P. & Cuthill, I. C. 1994b. Preference for symmetric males by female zebra finches. Nature, 367, 165-166.

Swaddle, J. P. & Cuthill, I. C. 1995. Asymmetry and human facial attractiveness: symmetry may not always be beautiful. Proceedings of the Royal Society of London Series B, 261, 111-116.

Swaddle, J. P. 1996. Reproductive success and symmetry in zebra finches. Animal Behaviour, 51, 203-210.

Swaddle, J. P. 1999. Limits to length asymmetry detection in starlings: implications for biological signalling. Proceedings of the Royal Society of London Series B, 266, 1299-1303.

Swaddle, J. P. 2000. Is fluctuating asymmetry a visual signal? In: Animal Signals-Adaptive Significance of Signalling and Signal Design in Animal Communication (Ed. by Y. Espmark, T. Amundsen & G. Rosenqvist). Proceedings Fifth

International Kongsvoll Symposium, September 15-20, 1998. Trondheim, Norway: Tapir Publishers. In press.

Thomas, R. J. 1999a. Two tests of a stochastic dynamic programming model of daily singing routines in birds. Animal Behaviour, 57, 277-284.

Thomas, R. J. 1999b. The effect of variability in the food supply on the daily singing routines of European robins: a test of a stochastic dynamic programming model. Animal Behaviour, 57, 365-369.

Tinbergen, J. M. 1987. Costs of reproduction in the great tit: intraseasonal costs associated with brood size. Ardea, 75, 111-122.

Verrel, P. 1982. Male newts prefer large females as mates. Animal Behaviour, 30, 1254-1255.

Waas, J. R. & Wordsworth, A. F. 1999. Female zebra finches prefer symmetrically banded males, but only during interactive mate choice test. Animal Behaviour, 57, 1113-1119.

Waser, P. & Waser, M. S. 1977. Experimental studies of primate vocalization: specializations for long distance propagation. Zeitschrift für Tierpsychologie, 40, 37-52.

Witter, M. S. & Cuthill, I. C. 1993. The ecological costs of avian fat storage. Philosophical Transactions of the Royal Society of London B, 340, 73-92.

Witter, M. S., Cuthill, I. C. & Bonser, R. H. C. 1994. Experimental investigations of mass-dependent predation risk in the European starling, Sturnus vulgaris. Animal Behaviour, 48, 201-222.

Workman, L. & Andrew, R. J. 1986. Asymmetries of eye use in birds. Animal Behaviour, 34, 1582-1584.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Zann, R. A., Morton, S. R., Jones, K. R. & Burley, N. T. 1995. The Timing of Breeding by Zebra Finches in Relation to Rainfall in Central Australia. Emu, 95, 208-222.

Chapter 2. General & Technical Methods

Introduction

A precursor to the experimental work performed throughout this PhD project was the design and creation of hardware and software involved in the collection of data and control of the experimental environment. I began this project with virtually no computer programming experience although I did have a reasonably good knowledge of electronics and mechanics. The first year of my project was almost exclusively devoted to the designing, building and perfecting of the equipment and software that I used to collect data, transform data and control the lighting/temperature environment of the laboratory. Every item of software and hardware described in this chapter was exclusively and solely written or built by the author, without external assistance.

In this chapter I will discuss the form and function of the 'microswitch perch' units that were used by male zebra finches as a forum upon which to perform their stereotypical display hopping behaviour. From there I will go on to describe the electronic balances that were used to collect body mass data, their association with the microswitch perches, and how data from the balances and perches were routed to the laboratory computer via expansion cards.

The acquisition, processing and storage of these data were mediated by a single software package named 'DataLog', which I custom wrote for the task. At the outset of the project it was difficult to foresee the exact detail of data I would wish to record and how I would wish to analyse it. To overcome this potential difficulty I designed the software to record all the information being made available to it. A consequence of this was that the data files that were generated were very large and often amounted to several thousand lines of text per bird per day. To assist in the analysis of these data, a separate piece of software called FileParse was developed, which contained seventeen parsing routines, each of which performed a specific and complex function. One benefit of using a post-hoc tool such as this was that no compromises had to be made at the data logging stage. Therefore, the full depth of information was always preserved, even if a condensed form of the data was used in the statistical analyses.

The lighting and temperature of the experimental environment was identified at an early stage in the project as an area that required to be precisely controlled and monitored. A system, controlled by the computer, was developed to allow the light level, ventilation and temperature of the room to be manipulated in a predictable fashion. The software, named DawnDusk, and the lighting arrays, will also be covered in this chapter.

Microswitch Perch Units

The concept of using perches connected to switches to record the movements of zebra finches is not new, having been employed by other experimenters (e.g. Swaddle & Cuthill, 1994a, 1994b). The perch units used in the present work were engineered using carefully chosen materials to create reliable, fault-free and low-maintenance devices. At the outset of the project the microswitch hop perches were a stand-alone unit located in the display section of the experimental cages (adjacent to the stimulus compartment) (figure 1). The electronic balance was associated with the feed perch such that body mass data were captured during feeding bouts, not during display bouts, and information on the temporal organisation of feeding was also inherent in the balances' output.

This first design of hop perch unit was fully encased within a plastic instrument box making it relatively bulky and heavy, although at this stage neither of these factors were particularly important (figure 2). The pushrods that actuated the switches were initially made of an extruded fibreglass rod material that was chosen for its strength and light weight although it did have certain shortcomings (see later). Long lever sub-miniature microswitches (RS Components no. 159-4657: RS Components Ltd., Corby, Northamptonshire, UK) were used because of their low actuating force and long mechanical life characteristics. By varying the distance from the pivot point of the microswitch's lever to the base of the perch pushrod it was possible to tune the perch to respond to very precise minimum forces. In this case the pivot point/pushrod distance was set at 35 millimetres, so that the switch was closed by forces as low as 0.098 N (10g). This allowed the perch to be activated by the lightest zebra finches while still maintaining sufficient resistance to cause it to rebound (switch off) after a bird hopped off.

After a period of experimental recording it became clear that the birds were spending the majority of their time on the hop perches and making only brief and infrequent visits to the feed perch. As the electronic balance was situated beneath the feed perch this meant that very few body mass records were being captured. The lack of mass data posed problems in terms of the type of analyses that could be performed. At first glance the solution to this problem seemed to be to locate the electronic balance underneath the hop perches. This approach was not without difficulties however, as the hop perches were quite sizeable and weighed around 400 grams, about 65% of the total weighing capacity of a balance. In addition there was the problem of the cable running from the hop perch. Although fairly lightweight (4mm diameter) the presence of the cable had an extremely large influence on the forces transmitted to the load cell in the balance, causing the reading to vary by between tens and hundreds of grams. Overcoming these difficulties meant stripping down the hop perch to minimise its weight without compromising its functionality, and also devising a method of running five (i.e. four lines plus a common wire) individual electrical connections away from the hop perch without hampering the correct operation of the balance.

By removing the casing from the hop perches and retaining only essential parts I successfully shed nearly 250 grams from its weight and roughly halved its physical size. The problem of running connections from the hop perch was solved by running five individual filaments of hair-breadth copper wire from the perches to a circuit board glued to the top of the balance. From here the original cable took over the connection back to the computer. The extreme delicacy of these wires meant that there was no restriction of vertical movement of the hop perch relative to the balance whatsoever. However, the wires were very prone to breakage and considerable care had to be taken to avoid rotating the hop perch relative to the balance, for example when reaching into a cage to catch a bird. A metal guard was glued in position to prevent a bird from disrupting the fine wires or entangling itself in them and as a result there were no cases where a bird caused a breakage.

An additional improvement was made to the microswitch perch units at this time. The extruded fibreglass material used for the pushrods was somewhat prone to sticking, partly because it presented a high friction surface to the PVC bush within which it

travelled. Moreover, its small diameter (3mm) made it more prone to sticking when transmitting non-vertical forces (i.e. when the bird was not central on the perch the force through the pushrod was not acting in a perfectly vertical manner). Therefore it was replaced by 6 millimetre diameter PTFE (Teflon) rod (RS Components 680-612) , which was chosen for its excellent low friction properties. The PTFE running in the PVC bushes proved to be a highly reliable system that gave very few problems except occasionally when blocked by detritus.

Having made the association between the balance and hop perch units there was a requirement to devise an alternative means of recording feeding bouts. This was achieved by building separate microswitch feed perch units each incorporating a single switched perch. The design of these was essentially the same as the hop perches albeit with a longer pushrod. Although the perch itself functioned flawlessly, the information on feeding bouts was generally poor and variable, the reason being that birds often chose to land on the rim of the feed dish instead of on the perch that was provided. There was little that could have been done to prevent this from happening. The final experimental cage configuration that was used in all studies in this thesis is given in figure 3.

Electronic balances: recording body masses

The balances used in this project were Sartorius PT-610s (Sartorius, Epsom, U.K.) fitted with optional serial data interfaces. These balances had a total weighing capacity of 610 grams to 0.1 grams resolution. The serial interface was configured such that the balance outputted approximately five mass readings per second, in synchrony with its own LCD display. A user configurable option is to set the serial port only to output data when the reading is stable. This proved to be unreliable as what the balance considered stable often still reflected a bird that was shuffling around. The preferred option was to make the balance output data continuously, and allow the computer to determine (see later) what represented the true mass of a bird and what did not.

The top pan of the balance was removed and the hop perch unit glued directly onto a plastic sleeve that sat on the load cell. The whole balance/perch combination was then

placed inside the experimental cage and the result was both unobtrusive and placed the hop perches at an ideal height from which the male zebra finches could display.

DataLog software

The data-logging program had to acquire and store data, simultaneously, from eight electronic balances and forty individual microswitch perches. The first step was to decide on a programming language to use and learn. I opted for Borland Delphi (ultimately version 5) which is a visual development tool for the Microsoft Windows environment. The word 'visual' refers to the manner in which the user interacts with the finished application. As is typical for most Windows applications the user is able to steer the program by means of pointing and clicking buttons and other objects using a pointing device. Delphi is a well-supported and widely used language that is generally regarded as surpassing Microsoft's own visual programming language, Visual Basic. Due to the amount of time that was to be invested in learning how to program, it seemed eminently sensible to learn a language that would continue to be useful above and beyond the scope of my PhD project, and this has proven to be the case.

The key elements to a visual language are the objects that appear on screen (and occasionally some that do not) and the programming code associated with them. Different objects are susceptible to particular actions, for example clicking or dragging, or alternatively system generated actions such as a timer event. Each event has an 'event handler' linked to it and the programmer can insert code into this handler such that the code is run each time the event occurs. The first and probably biggest hurdle of all, was figuring out how to interface all the hardware with the computer and then how and where to access the information in a useable form. For many weeks my key target was to be able to see a reading from an electronic balance appear on my computer screen. Although the ultimate goal was far more complicated, I knew that once I had a means of getting a mass reading into a variable in computer memory, I could realistically create the finished program.

DataLog software: body mass data

Having eight balances all outputting data via serial ports posed a problem as far as interfacing them to the computer was concerned. One possible solution was to use a multiplex device. This would have involved connecting all eight balances to the multiplex device that would then be linked back to a single serial port on the computer. The computer or multiplexer would then execute a continuously looping scan of all eight balances and extract a reading from each in turn. This solution was not favoured as it would have resulted in $^{7}/_{8}$ of the available data being lost.

The alternative was to create eight serial ports on the computer. Usually a personal computer is supplied equipped with two serial communications ports; COM1 and COM2. To create eight ports I used a serial communications expansion card (Amplicon PC231: Amplicon Liveline, Brighton, UK). Once installed and configured the computer had available its standard COM ports plus an additional eight (COM1 through COM10 in total). An eight way splitter box (equipped with eight 9 pin D-connectors) was built to connect to the 78 pin connector on the back of the expansion card, and each of the eight balances was connected via its own cable to the splitter box.

In order to address these ports from the software I used a serial communications component designed to operate within Delphi 2 (VSSComm32). This non-visual component mediated the communications between my finished program and the serial ports. Eight of these components, each corresponding to a separate COM port, were required. This component has an event handler called 'OnReceiveData' that fired each time a complete serial data word was received from the balance. It was into this event handler that the code to acquire and record the balances' output was inserted. For an example of this code see lines 234 to 261 in Appendix 1, which is a complete listing of the DataLog program.

A schematic diagram showing the execution of code within an 'OnReceiveData' event handler is given in Figure 4 in the form of a flowchart. The first task for the program was to read the balance output from a COM port's memory buffer into a local variable. In order for the program to determine whether the balance output was stable or not it kept a rolling record of previous mass values. To do this in practice only the

penultimate mass value needed to be recorded and a count was made of the number of consecutive values that had been the same. Various measures of stability were tried and tested but ultimately I settled for one where 8 consecutive identical values were necessary before the software treated a balance output as being truly stable. This corresponded to a reading having been stable for approximately two seconds. Each time the balance output changed, the 'counter' variable was reset to zero and only when it was greater than or equal to 8, did the program treat the reading as having stabilised.

Once the code had determined that stability had been reached, the current balance reading was stripped down, character by character, to just the key numerals (discarding other non-numeric characters that the balance unavoidably sent). The program then determined whether the stable reading it had captured corresponded to the mass of a bird plus 'tare' (bird on perch) or purely 'tare' (no bird on perch). The balances could drift by several grams over the course of an experiment and they were rarely manually zeroed as this was unnecessary and could cause transient problems for the software routine involved in discriminating mass output.

The rule to determine if the reading was 'bird' or 'tare' simply involved checking whether or not the reading was less than +5g and greater than –5g. If this test returned true then the current reading was treated as 'tare' and the routine reset such that it waited for the balance to output a new value. If the test returned false the program checked whether the current value, less the currently stored 'tare', was greater than 10g (i.e. was a realistic zebra finch's mass). If this test returned true then the value was written to the appropriate textfile along with the current time of day.

Many different discrimination rules were employed throughout the course of the project but the one described above was one of the least complicated but also one of the most reliable. It makes the assumption that an adult zebra finch will never weigh less than 10 grams (in practice the lightest individual I came across was slightly heavier than 12 grams) and it also relies on the balances never drifting more than ± 5g . In practice the amount of drift never approached this value and a daily check ensured that those balances that had drifted noticeably were reset to zero. Finally, the routine assumed a reading was stable when the same value had been output eight consecutive times

(equating to approximately two seconds). This was a conservative rule and therefore ensured that high or low mass outputs caused by birds moving on the perch were never erroneously recorded.

Records were written to the appropriate textfile in the format 'time of day in milliseconds, mass of bird in grams (to an precision of 0.1g) & time of day in format 'hh:mm:ss'. The time of day information was written to file twice as the first (milliseconds) format was more convenient for post-hoc processing routines, and the second (hh:mm:ss) format was more readable if I required to browse a file manually. Each experimental cage had a unique textfile created automatically every day, the filename of which incorporated the identity of the cage and the date. This greatly simplified the organisation, compression and analysis of the data being generated.

DataLog software: hop perch data

The other main type of information that had to be accessed and logged was the data from the hop perches and feed perches. Detecting the on/off status of a microswitch perch was achieved using a 48 line digital input/output expansion card (Amplicon PC214E) which was installed in the computer. Each of the 48 channels had two possible states: either high (binary 1) or low (binary 0). In electronic terms, a high state occurred when the input was between +2.2 and +5.3 volts, and a low state occurred between –0.5 and +0.8 volts. By using the +5.0 volts supply from the computer it was possible to wire the perches such that a bird landing on one caused the corresponding input channel to go from low to high (+5.0 V), as shown in figure 5. Each individual hop perch and feed perch unit was wired back to a splitter box, and a multi-way cable (32 lines for hop perches and 8 for feed perches) was run from there back to the 78-pin connector on the rear of the expansion card.

The status of switches was reflected by the output from the registers of the I/O card, which was available directly to the central processing unit (CPU) of the computer. The software read the data in the form of five individual data words that were in decimal format. These decimal strings were converted using a custom-written software routine, to binary equivalents containing eight characters, each of which represented the on/off status of a particular switch.

Although the status of switches was changing constantly this was not used to trigger an event handler within Delphi. Instead, the code that read the switch statuses and logged perch movements was inserted within the 'OnTimer' event handler of a Timer component, which meant that all perches were queried at a preset time interval. This timer was set to fire every 30 milliseconds and each time it fired it scanned the status of each of the 40 perches. Nearly 450 lines of code resided within the 'OnTimer' event, therefore every time the event occurred there was a significant burden on the processor. 30 milliseconds was chosen as the period for the timer as a compromise between limiting the workload of the processor while preserving the accuracy of the data. The maximum error that could have occurred in the recording of the time spent on a perch was plus or minus 60 milliseconds and this was regarded as being completely acceptable. Whenever a switch was activated by a bird landing on a perch the software noted the 'On' time such that when the bird left that perch again the time spent on it could be calculated by simple subtraction. This information, together with the identity of the perch and the time of day, was written to the appropriate textfile. In the same way as for the mass data, a unique textfile was created every day to record each birds' use of the different hop perches and feed perch. A schematic diagram showing the execution of code within the 'OnTimer' event handler is given in the form of a flowchart (Figure 6). An example of a single instance of this code can be found in Appendix 1, lines 584 to 606 (note that there are in total 40 instances of similar code, corresponding to the 40 perches).

A screenshot from the DataLog program is given in figure 7, in order to illustrate the graphical interface and the information that it presented to the user.

FileParse software development

This piece of software was developed as an aid to the organisation and analysis of mass and hop data. The textfiles that contained the raw data were often very large with thousands of lines of information, and it quickly became apparent that I required a means of condensing them into a standardised and more useful format. The FileParse program was gradually expanded over the course of my project, to the extent that it

finally contained seventeen discrete routines forming in excess of 2,200 lines of programming code in total.

Each routine performed a very specific function. The majority read through a data file and then wrote the condensed information into a new file with its name deriving from the source data file. In these cases the original source data were always preserved so that the full depth of information was never lost, should it have been required again at a later date. In this section I describe the key routines that were written, and their functions. The most important routines were generally also the most complex and I begin with these and discuss less important or redundant routines last. The headings used throughout this section relate to the abbreviated names of the routines as shown on the control buttons (see figure 8).

'*Transform File- Perch Data*' routine

This was one of a number of key functions of the FileParse program. This routine condensed the hop perch information for one bird (sometimes comprising more than 17,000 records daily) into the total number of perch movements during a minimum of 24 and maximum of 144 periods covering the whole day. The temporal distribution of the periods was user defined. Throughout my work I have typically defined a day as comprising 48 half hour intervals. These intervals were set up using the panel located at the bottom right of the FileParse window ('*Set up hourly time intervals*'). The user could move through each hour of the day (numbered 0 through 23) to confirm or change the number of intervals into which that hour was divided. To change the value for any given hour the user simply typed in the new value and clicked the '*Select time interval for each hour*' button. Once all the values were set up as desired the user clicked the '*Set time intervals as selected*' button and the new values were then stored permanently until edited again.

So that the user did not have to redefine the desired time intervals each time the program was run, I employed a programming device called an 'ini' file (.ini file extension). This is a Windows feature whereby information is stored in a special textfile on the hard disk. The program knows the location of the ini file and retrieves information by specifying the value it requires and the grouping into which it falls. In

this case there were two groups of information; 'TimeIntervals' and 'SecondGaps'. The first was a permanent record of the number of intervals for each given hour of the day and the second was a reference list giving the time in seconds from the beginning of the day when each interval started and ended (calculated according to the specified number of intervals in a given hour). Whenever the user selected new time intervals the ini file was automatically rewritten to reflect those changes.

A relatively late addition to this (and other) routines was a feature whereby the program could be set to parse all files in a given folder whose date identifier fell within user defined start and finish dates (see bottom-left panel of fig.8). This meant that the perch data files (as an example) from a complete experiment could be placed into their own folder and a start and finish date keyed in. When the button was pressed the program automatically sought a file for cages 1 to 8 on each of the specified days. Wherever a file or group of files did not exist they were skipped over until the program found the next one that did exist. The code that carried out this task was complicated. To enable the program to sequentially move from day to day (involving transitions from month to month and also from year to year) it had to be aware of the numbers of days in each month and when to click over to the next year. This piece of code was written in the format of a function, which is a piece of code that carries out a calculation on a received value or string and returns a value accordingly. In this case the function *NextDayOfYear* received a date in the format 'dd mm yy' and it calculated the following day in the same format.

Prior to starting the Transform File-Perch Data routine, the user browsed to the folder containing the data that required to be condensed. When the desired folder was selected, and the start and end dates were defined, the user clicked the '*Transform File-Perch Data*' button to begin the process. The code began by looking for a file for cage 1 on the start day that was selected. If this file did not exist the code then looked for a file for cage 2 on the start day and so on until the first file in the sequence was found. The code continued in that manner, scanning for files for cages 1 to 8 for all of the specified days. When a targeted file was found to exist, the main body of the routine began. The code firstly made a preliminary reading of the whole file to determine the number of lines it contained and stored this value. Having done this the program referred to the .ini file and requested the start and finish times, in seconds, of the first interval. The code then

began at the start of the datafile again, this time reading in the time of day in seconds for the line that corresponded to the time at which the first single hop began. A check was made to see if this time fell within the start and end times of the first interval. If it did not then the program wrote a value of zero to the results file for the first interval, and requested (from the ini file) the start and finish times for the second interval and again checked to see if the current perch movement fell within that interval. This process carried on until the interval for the first perch movement was found.

At this point a counter variable was incremented by one and the program proceeded to read in the next line from the data file. The time was once again checked against the start and finish times of the current interval, and so on. The program totalled the number of hops for the given interval and when it next read a time that did not fall within the current start and finish times, the code wrote the total number of perch movements for the given interval into the results file and it clicked forward to the next interval as defined by the .ini file. The code continued in this manner until each line from the data file was read and a total number of perch movements for each interval had been written to the results file. Once this was achieved the results file was closed, all counters were reset to zero (as applicable) and the program moved on to the next data file in the sequence. Each results file was uniquely named as a derivative (prefixed by 'Modi') of the data file that spawned it, thereby keeping its unique date and cage number identification tags.

In summary, by clicking a single button, it was possible to quickly and conveniently condense any number of perch data files into a compact and readily analysable format, giving measures of the rate of display hopping activity for specific intervals covering a whole day. A schematic diagram showing the execution of code within the '*TransformFile-PerchData*' event handler is given in the form of a flowchart (Figure 9) and the code for this routine can be found on lines 480 to 613 of Appendix 2 (which lists the complete FileParse program).

'*Transform File- Mass Data*' routine

This was the second key function of the FileParse program. This routine was designed to condense the body mass information into a series of mean values corresponding to a minimum of 24 and a maximum of 144 periods covering a whole day. Typically there were about 300 to 600 mass records stored for a given bird in any one day. The amount of information stored was dependent on the amount of time spent on the hop perches and the activity of the bird. A more active bird typically generated fewer stable mass readings. In exactly the same manner as for the '*Transform File- Perch Data*' routine, the distribution of the time intervals over which mean body masses were calculated was user defined. In the case of the mass data it was decided that each day should also be sub-divided into 48 half hour intervals.

As with the perch data parsing routine, this routine also automatically parsed all files within a chosen folder provided that they fell within pre-selected dates. The main body of the mass file parsing routine began by counting the total number of lines in the current data file. The code then returned to the beginning of the data file and read the first line into memory. The standard mass data file contained three elements per line, namely; the time of day in milliseconds, the mass in grams and the time of day in standard 'hh:mm:ss' format. These three elements were separated by commas and it was these commas that the routine first sought to locate, so that it was able to determine which characters in the string related to the mass value and which to the time of day in seconds. These digits were then sequentially read into the appropriate variables.

The routine then referred to the 'ini' file and read in the start and end times for the first interval. If the time reading from the current line of the data file fitted into the first time interval then the mass reading was added to a variable that kept a running total of mass for that interval. At this time a counter variable was also incremented by one. Once all the mass readings for a given interval had been extracted, the running total mass value was divided by the number of readings that contributed to it to generate a mean value for body mass, which was stored to the results file.

In summary, by clicking a single button, it was possible to quickly and conveniently condense any number of mass data files into a compact and readily analysable format,

giving measures of the mean body mass for specific intervals covering a whole day. A schematic diagram showing the execution of code within the '*TransformFile-MassData*' event handler is given in the form of a flowchart (Figure 10) and the code for this routine can be found in Appendix 2, lines 294 to 469.

'*Combine Files*' routine

This routine saved an enormous amount of time in processing all of the data files. Having been parsed, all data were imported into Microsoft Excel where they were organised and graphed as required, prior to being exported to a statistics package for analysis. Opening a textfile into Excel was a cumbersome task requiring the operator to specify if and how a file was delimited before the application would open it. Each experimental cage generated three (hop perch file, feed perch file and mass data file) separate data files each day equating to twenty-four data files in total for each day that an experiment ran. Prior to devising this '*Combine Files*' routine it was necessary to individually open/import each file into Excel before embarking on a very laborious cutting and pasting operation to combine them. I therefore devised this novel routine to allow me to combine all the textfiles for a given experiment and import them into Excel in one single operation, so saving time. The manner in which I required to combine the files is illustrated in figure 11.

This process required to be iterative such that multiple files (which, having already been subject to a post-hoc process, were of equal length) could be combined on a row by row basis. The routine that I created allowed the user to specify the name of the (final) results file. Having done this it was a simple matter of browsing to the first file that was to be included in the results file and clicking the '*Combine Files*' button. This procedure could be repeated any number of times. The routine worked with three open textfiles at any time. One of these was the file currently being added, one was the results file and the final one was a temporary read/write file. The routine began by reading in the first line from the results file into memory as a string variable. If the routine was being run for the first time for a specific results filename then the results file would have been initially empty, alternatively it would have contained some text. Having read from the results file, the routine then read the first line from the file that was to be added and placed it in memory as a string variable. These two strings were then concatenated

(separated by a semicolon so that Excel could recognise the breaks) and this resultant string was written into the temporary file location.

This process was repeated until all the lines of the file to be added had been read. At that stage the file being added and the results file were unchanged from the start of the routine, however, the temporary file contained the cumulative data from those two. The order in which they were added was the results file data followed by the data from the file to be added (i.e. the last data to be added was to the right). The routine then cleared the results file and opened it ready for writing. Each line from the temporary location was read firstly into memory and then written over into the results file such that a copy of the temporary file was created in the location of the results file. When this process was complete the temporary file was cleared and closed and the results file was saved and closed.

The process was then complete with the desired file having been added to the cumulative results file. There was no practical upper limit to the number of times this process could be iterated such that all the data for a given experiment could be accumulated in a specific order in a single textfile for extreme ease of import into Excel or another application. A schematic diagram showing the execution of code within the '*Combine Files*' event handler is given in the form of a flowchart (Figure 12) and the code for this routine can be found in Appendix 2, lines 965 to 1008.

'*Mass Gain Per Interval*' routine

This routine calculated the average rate of mass gain (either positive or negative) over every half hour interval throughout the day. The output generated by this routine for various files was therefore uniform in length with an equal number of comparable data points, which simplified the analysis process.

Since the temporal arrangement of mass records was essentially random, the routine required to seek the nearest start and finish mass, and start and finish time to suit the interval for which a mass gain rate was being calculated. This was achieved by the program sequentially reading through the raw data file and checking when the time difference between the time from the current line and the reference time for the current

interval, flipped from positive to negative. The first 'time from file reading' to give a positive difference and the last to give a negative difference were the two closest to the reference time in question. By comparing the absolute difference between the two nearest times and the reference time the closest was found, and it was this value that was used to evaluate the mass gain rate for the interval in question.

The change in mass and the exact difference in time between the measurement points at the beginning and end of the interval were calculated and from this the rate of mass gain was found. The information written to the textfile was the current interval end time, the actual end time (i.e. the time of the nearest record to the nominal end time) and the mass of the bird at the actual end time. A schematic diagram showing the execution of code within the '*Mass Gain Per Interval*' event handler is given in the form of a flowchart (Figure 13) and the code for this routine can be found in Appendix 2, lines 1196 to 1328.

'*Rate Of Mass Gain*' routine

This routine is similar to that described above except that it calculated the rate of mass gain between each mass reading and the subsequent reading. Whereas the standard parsed mass data allowed the overall trajectory to be viewed, the information created by this routine allowed the rate at which mass was being gained or lost to be considered throughout the day.

The routine read through the raw mass data files in the usual sequential manner, extracting the current mass and time. The code kept a rolling record of the previous mass and the time of day associated with it. Using this information the routine computed the change in mass over a given change in time and for these values a value for the rate of mass gain was evaluated. This value was written to file along with the time of day to which it related.

The data generated by this routine were not readily analysed, as the times of day at which mass gain rates were calculated, and the number of records per bird or per day, varied from file to file. As a consequence the data were cumbersome to graph or compare and they had limited value. This is in contrast to the '*Mass Gain Per Interval*'

routine described above. The code for this (the '*Rate of Mass Gain*' routine) routine can be found in Appendix 2, lines 618 to 689.

'*Prop Hops Per Perch Per Day*' routine

This routine was designed to compute the number of hops (expressed as a proportion of the total) on each of the four display perches for a given bird on a given day. This allowed comparisons to be made between treatments as to differential perch usage.

This routine condensed all the perch data from an entire experiment into a single textfile that was then easily imported into Excel. Having selected the folder containing the raw perch files, and with the required date range defined, a single button click automatically accessed all the data and calculated the perch usage information. The output file contained a single line per cage per day, listing the proportion of total hops made on each of the four perches followed by the cage identifier followed by the date. This format, and the fact that all the information was now in one file, made the summarised data very easy to work with.

The routine worked by sequentially reading through the perch data files and extracting the perch identifier information from each line. A running total was kept for each file of the number of instances on which each perch was used. When a file had been completely read the proportions were calculated and written along with cage identifier and date to the results textfile. Any number of raw data files could be condensed into a single results file by a single button click. While there is no schematic diagram of the flow of execution for this routine, the code can be found in Appendix 2, lines 1511 to 1600.

'*Condense Perch TB*' and '*Condense Feed TB*' routines

In addition to the individual hop perch data that was gathered for each cage, a single file was written at a specified interval (usually every ten minutes) which recorded perch time budget information. For a given time period, the DataLog program recorded the time spent by each bird on its hop perches and feed perch, the proportion of time spent

on the hop perches or feed perch and the total number of perch movements for each type in that period.

The design of the DataLog program meant that excess information was being written to the time budget files. This routine simply removed this superfluous data and wrote a more condensed results file that contained only the proportion of time spent by each bird on the display/feed perches and the time interval to which that applied. There are no schematic diagrams of the flow of execution for these routines, however the code can be found in Appendix 2, lines 1358 to 1430, and lines 1434 to 1506.

'Condense Every Bal Output' routine

This routine was used to remove lines from the mass data files that did not represent stable balance output. For a period during its development the DataLog program was recording every balance output without applying any corrections to the incoming data. This is in contrast to the initial and latter designs in which the DataLog program corrected for drift before writing 'clean' mass readings to the text file.

During the time when the DataLog program was recording every balance output, a running total of the number of consecutive identical outputs was kept and only when the output changed did this trigger the program to write the output to the results file. Each line of the results file had a value denoting how many times that particular output occurred. In order for an output to be assumed to be stable, there had to be more than 20 repeats, which was in fact a highly conservative figure. This was based on a period of 4 seconds during which the output (5 outputs per second) remained stable. This routine simply read through each textfile and extracted the value for the number of repeats that occurred. Only those lines with more than 20 repeats were written to the results textfile. Note however that the results file did not contain corrected mass readings, rather, it contained all stable values whether they represented drift or bird mass plus drift. A separate routine was then used to calculate actual bird mass values.

The DataLog program was not used in this manner for a long period as the output files were unnecessarily cumbersome and required excessive post-hoc processing. The program was modified, based on an improved version of the original design, such that

all drift corrections were carried out prior to any data being written to file. There is no schematic diagram of the flow of execution for this routine, but the code can be found in Appendix 2, lines 693 to 767.

'*Correct Mass For Drift*' routine

This routine was designed as an adjunct to the above routine to remove the effects of drift from the mass data files. This was only necessary for a limited period of time during which the DataLog program was configured to write every balance output to file. The routine sequentially read through the files containing every balance output and extracted the mass and time information for each line. By keeping a rolling record of previous masses, the routine was able to use a standard set of rules to determine whether the present mass reading represented drift, or bird mass plus drift. Corrected masses were then written to the results text file. The code can be found in Appendix 2, lines 853 to 958.

This adjustment is now made by the DataLog program prior to writing any data to file.

DawnDusk Software Development

Due to the nature of the experiments, and the room in which they were taking place, it was regarded as important to be able to accurately control the lighting and temperature to which the birds were subjected (see chapter 3). The building where the birds were housed was light-sealed and as such all the ambient lighting was artificial. In such a situation it is not uncommon to have the room lights turned on and off, in unison, by an electro-mechanical timer. Although such devices permit accurate control over the light/dark photoperiod, they create a situation of increasing/decreasing light intensities completely unlike anything that would ever be encountered in nature. Given that some of the most interesting behaviour, in terms of courtship behaviour and the trade-off between this and foraging, typically occurs at dawn and dusk, it was apparent that some attempt to replicate natural conditions could be beneficial.

I decided that I would seek to have a slowly increasing and decreasing light level at dawn and dusk respectively, as opposed to an all-or-nothing lighting regime. The

obvious way of achieving this was, at first glance, to employ dimmer devices. There were two problems with this approach. Firstly, the majority of the ambient light was provided by fluorescent strip lights, and it is difficult and expensive to reduce their output reliably. Secondly, there was the problem of how to automate the function of a dimmer device.

The solution I chose was to construct an array of low-power incandescent light bulbs that were capable of being switched independently, such that the light level could be increased or decreased by turning lights on or off in sequence as appropriate. The switching of the lights was mediated by a reed-relay PC expansion card (Amplicon PC263) which provided 16 switched outputs. A linear array of 8 standard batten lamp holders, spaced at 200mm intervals, was mounted on a board, and each holder was equipped with a 60W tungsten lightbulb. These, together with 6 fluorescent striplights (58W each), and a pair of 300W halogen floodlights provided all the light within the experimental environment. Ten switching events acted on these light sources in the following order; each of the 8 tungsten lightbulbs (individually), then all six fluorescent striplights in unison, and finally both halogen floodlights also in unison. When switched in this sequence the pattern of light level was intended to mimic dawn, and the reverse sequence mimicked dusk (see chapter 3, figure 2 for details of the relative changes in intensity).

The PC263 relay card was only rated for switching loads of up to 15W, which was below the minimum requirement of most of the devices being switched. To overcome this I used intermediate relays on all those switching channels that were used. These intermediate relays were rated at up to 5A at 250 volts, and their switching coils were 12 volts DC. A separately cased bank of eight relays was used to switch the linear array lights with individually cased relays switching the fans, fluorescent strips, floodlights and sealed unit devices (fans and lights). These intermediate relays were themselves switched by a low voltage power supply routed through the PC263's relays. A major advantage and safety feature of this was that the 240 volts mains supply was kept isolated from the reed relay PC263 board/computer and that the multi-core cabling running from the rear of the relay board was carrying only low voltages.

The relay board was also used to control the temperature within the experimental environment. In order to achieve this, the laboratory computer had to be temperature-aware. This was made possible using a thermometer device (Pico Technology TC-08, Cambridgeshire, UK) which connected to the PC via one of its spare serial ports. This device had a capacity for the connection of up to eight thermocouples from which temperature readings were passed as serial data, and as such were accessible to software. In order to adjust the temperature, the relay board was used to switch a pair of extractor fans, which were located in either end of the building. Via the software interface, the user was able to define the required temperature. The maintenance of this temperature was partially achieved by the simple thermostatic controls on the heaters used in the room, but was further assisted by thermostatic control of the ventilation. If the temperature exceeded the desired level, the extractor fans were turned on, and being located high in the roofspace they quickly reduced the temperature by expelling hot air and drawing cooler air in from outside. Conversely, if the temperature fell below the optimum level, the fans were turned off and heat was conserved within the building. In addition, the PC263 board was employed to control the lights and ventilation in the sealed unit that was used to house birds whose sense of time was being desynchronised (see chapter 5 for details).

The software used to control the lighting and temperature within the experimental environment was referred to as DawnDusk. Although this application was also involved with temperature and ventilation, its main role (as its name suggests) was to control the lighting regime, and it is with this that I shall deal first. The user interface (figure 14) allowed the user to define the lighting regime and the associated parameters. At the highest level there was an option to select either a rectangular regime or a sigmoidal regime. A rectangular regime refers to a situation where all the lights came on and went off in unison, as would occur in a simple mechanical timer setup. If this option was selected then the only relevant parameters for user input were the 'Time of first light' and the 'Daylength in hours'. The sigmoidal regime was intended to replicate the natural exponential increase and decrease in light level associated with dawn and dusk. The user was again required to define the time of first light and the daylength. The time of last light was simply the time of first light plus the numbers of hours of daylength that was specified. The user also specified the required duration of the dawn and dusk

periods and this setting determined how long the sigmoidal period lasted (i.e. from the first switching event through to the final event).

There were also parameters associated with lighting events within what was termed the 'Sealed Unit'. The sealed unit refers to a specially constructed cage that was equipped with its own lighting and ventilation. The lights in this environment could be set to switch on or off independently of the rest of the experimental environment. The sealed unit was so-called as the front of the cage was blacked out with a heavy black polythene curtain. This allowed the birds in this cage to have their time-frames precisely manipulated independently of the prevailing time of day (see chapter 5).

When the program was first run or the dawn/dusk parameters were altered, the times of each switching event were placed in memory. The code that carried out all of the switching events was run twice a second and each time it ran it performed a check to see if any of the specified switching times matched the current real time. If the real time matched a specified time for a switching event then that event occurred and a line of text was added to a textfile to specify the identity and status of the light that was switched, and the time and date at which it occurred. This ensured that a permanent log was kept of the lighting conditions on any given day. For testing purposes, the software displayed the status of all of the lights and fans on screen, and allowed each channel to be manually overridden by an on-screen button click. The program also gave an on-screen display of the current temperature (for both the main room and the sealed unit). These were updated once a minute and at the same time both temperatures were written with times and dates to a textfile, so that a permanent record of temperature was available. The temperatures were compared against the selected cut-out temperatures and the sealed unit fans and main room fans were switched on or off as appropriate.

A full listing of the code for all the routines within DawnDusk is provided in Appendix 3.

References

Swaddle, J. P. & Cuthill, I. C. 1994a. Preference for symmetric males by female zebra finches. Nature, 367, 165-166.

Swaddle, J. P. & Cuthill, I. C. 1994b. Female zebra finches prefer males with symmetric chest plumage. Proceedings of the Royal Society of London Series B, 258, 267-271.

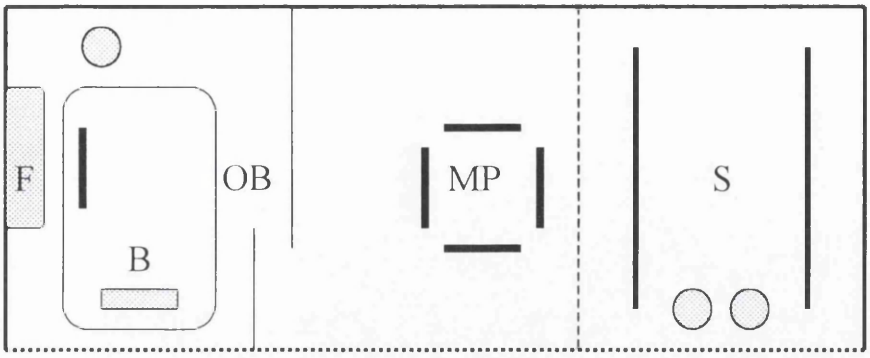


Figure 1. The original experimental cage design, as viewed from above. F = feed hopper, B = electronic balance mounted beneath feed perch, OB = opaque baffles, MP = microswitch perches, S = stimulus cage. Heavy bars represent perches, dotted lines represent wire cage front and Perspex panel separating courtship and stimulus compartments. Shaded circles are food and water dishes.

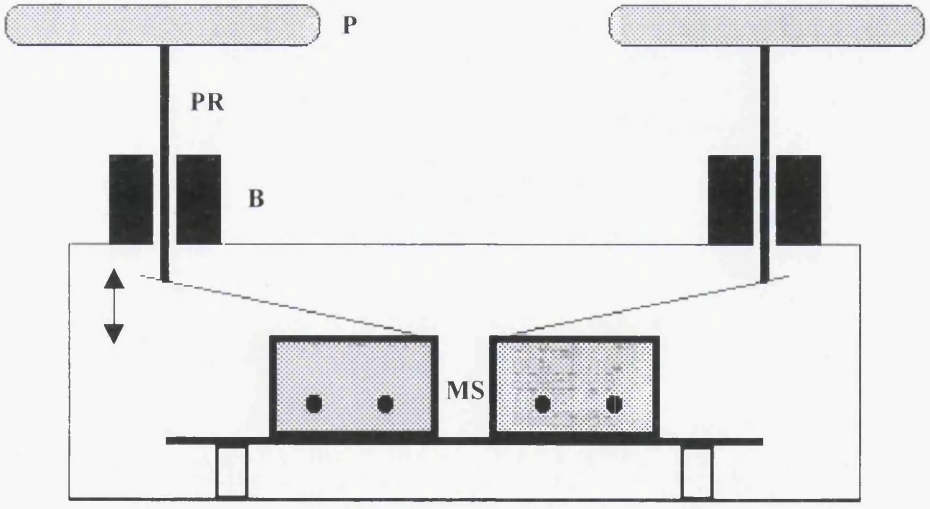


Figure 2. Side elevation of microswitch perch mechanism (boxed). P = perch, PR = Teflon pushrod, B = PVC bush, MS = long-lever PCB mount microswitch. Two out of four perches are shown for clarity.

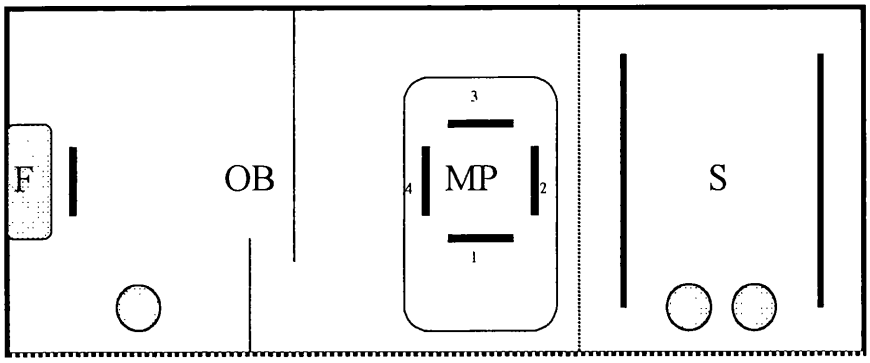
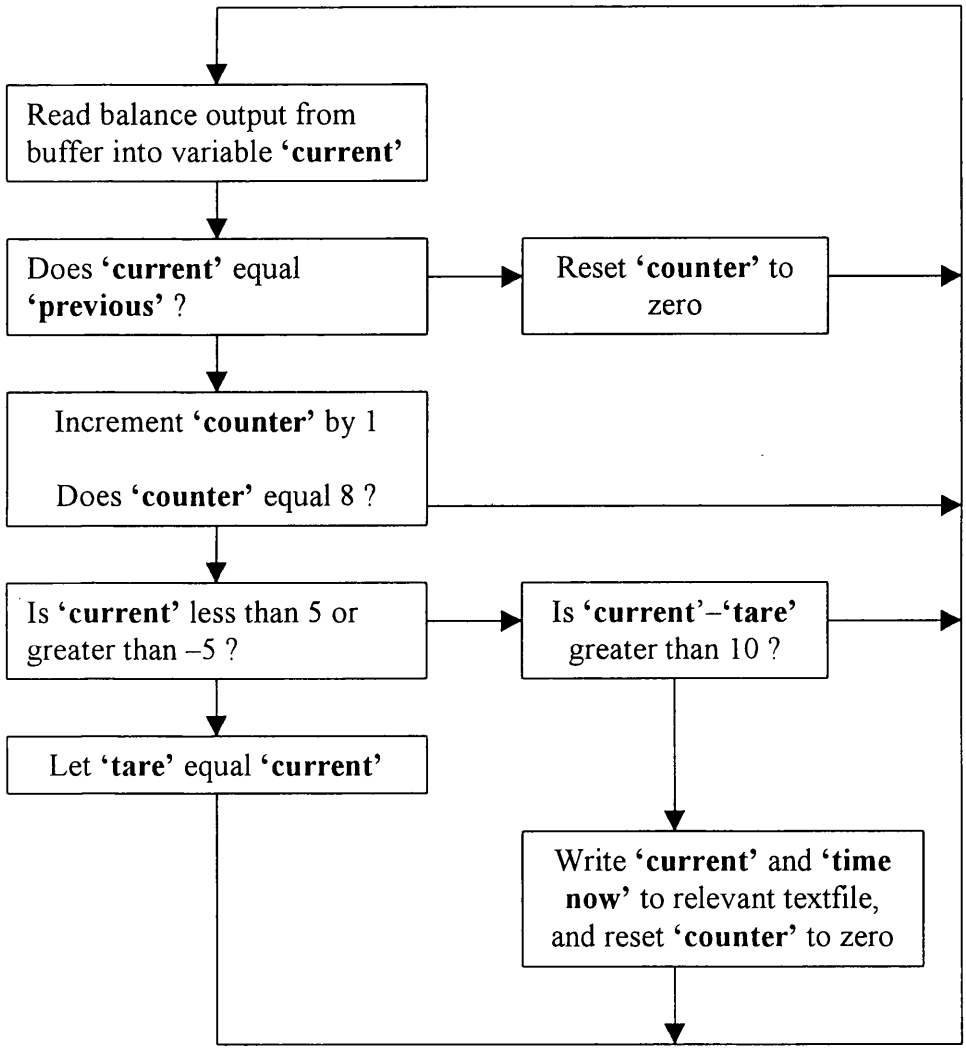


Figure 3. The revised experimental cage design, as viewed from above. F = feed hopper, OB = opaque baffles, MP = microswitch perches mounted on top of electronic balance, S = stimulus cage. Heavy bars represent perches, dotted lines represent wire cage front and Perspex panel or wire mesh separating courtship and stimulus compartments. Shaded circles are food and water dishes.

Figure 4. Flowchart diagram showing flow of execution for an '**OnReceiveData**' event handler within the DataLog program. For an example of the code for this event see Appendix1, lines 234 to 261.

Note: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right represents 'no'.



Variables:
'**current**' is most recent output from electronic balance
'**previous**' is the penultimate output from the balance
'**counter**' tracks stability by counting the number of identical consecutive readings
'**tare**' represents the present level of drift from zero in the balance's output
'**time now**' is the current time of day

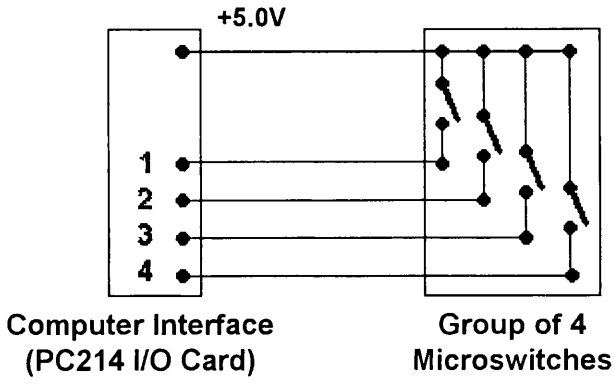
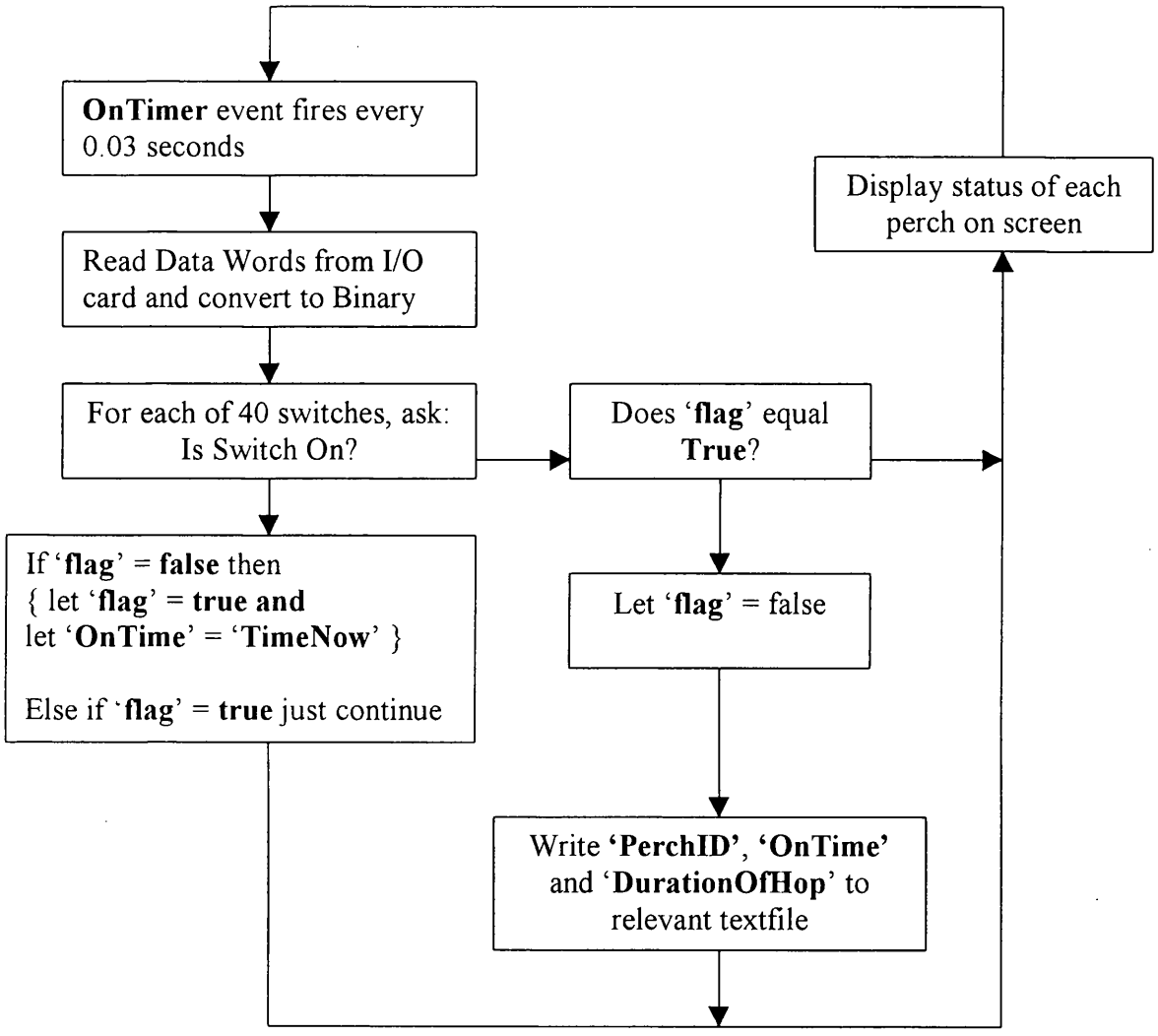


Figure 5. Schematic wiring diagram of a group of four microswitch perches. In total 40 input lines were required overall for the eight display perch units and the eight individual feed perches. A common wire carried +5.0V. Each time a switch was activated it set the corresponding input of the I/O card to 'high'.

Figure 6. Flowchart diagram showing flow of execution for an '**OnTimer**' event handler (within the DataLog program) which scans status of all microswitch perches. For an example of a single instance of this code see Appendix1, lines 584 to 606.

Note: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right represents 'no'.



Variables:

'**flag**' is a Boolean variable which shows True while a bird is alighted on a perch (i.e. during a hop) and reverts to False when the perch is released. It serves as a marker that a hop is ongoing.
'**TimeNow**' is the current real time of day
'**PerchID**' is the identifier of the particular perch. Within a display perch unit each of the four perches is uniquely identified.
'**OnTime**' is the real time at which a hop began (i.e. bird first alighted)
'**DurationOfHop**' represents the length of a hop in milliseconds. At the moment this variable is written it is equal to '**TimeNow**' – '**OnTime**' (i.e. the present real time minus the time at which the bird first alighted on the perch)

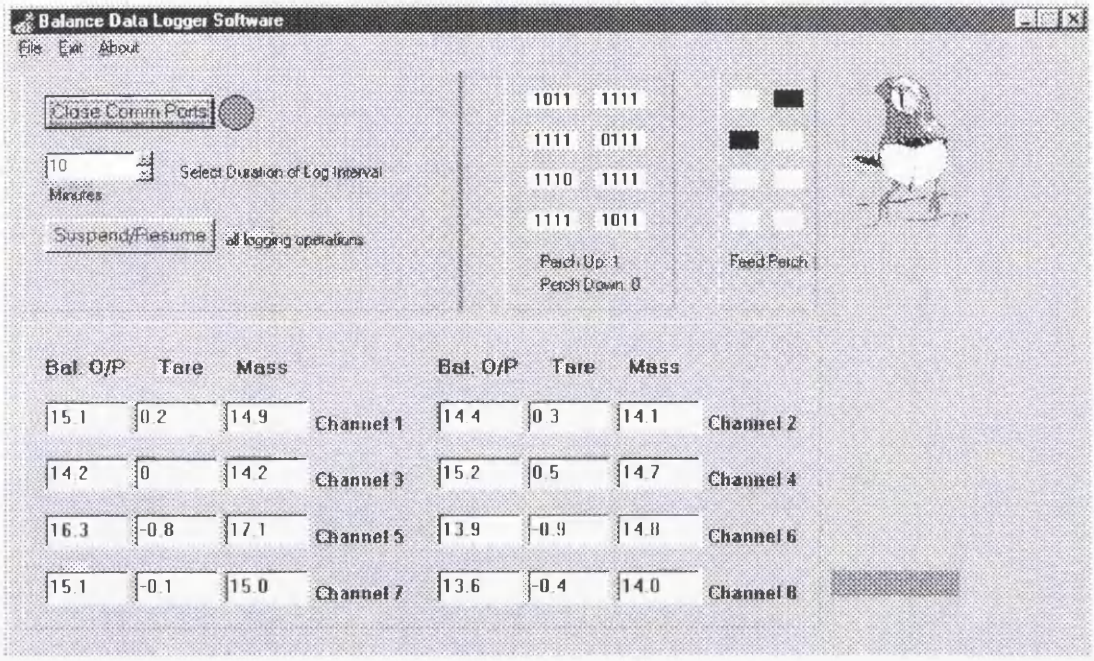


Figure 7. Screenshot of the DataLog program. Top-left panel allows logging operations to be started/stopped and periodicity of time-budget logging to be set. Top-middle panel shows real-time status of the 32 hop perches (in eight banks of four), top-right panel shows real-time status of the eight feed perches. Bottom panel shows real-time uncorrected balance outputs, extent of drift, and corrected mass value.

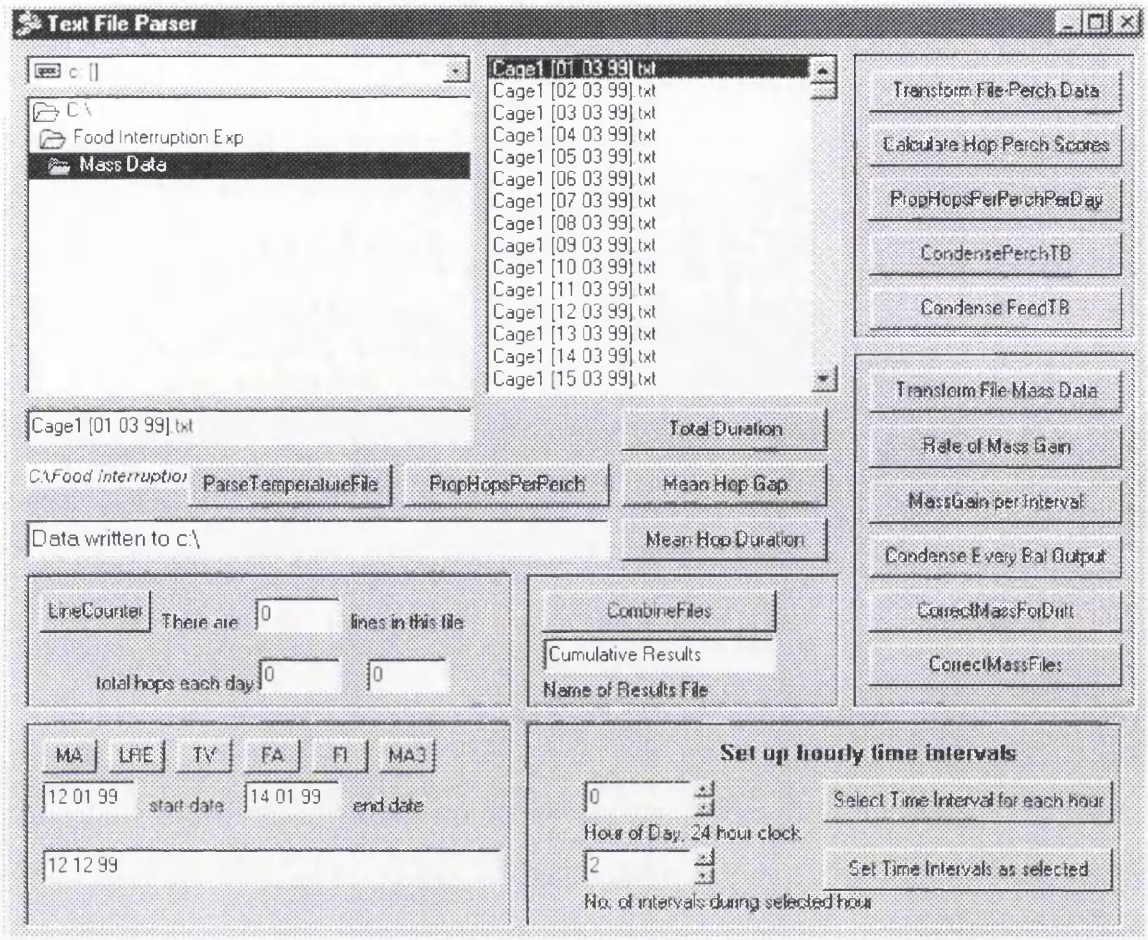
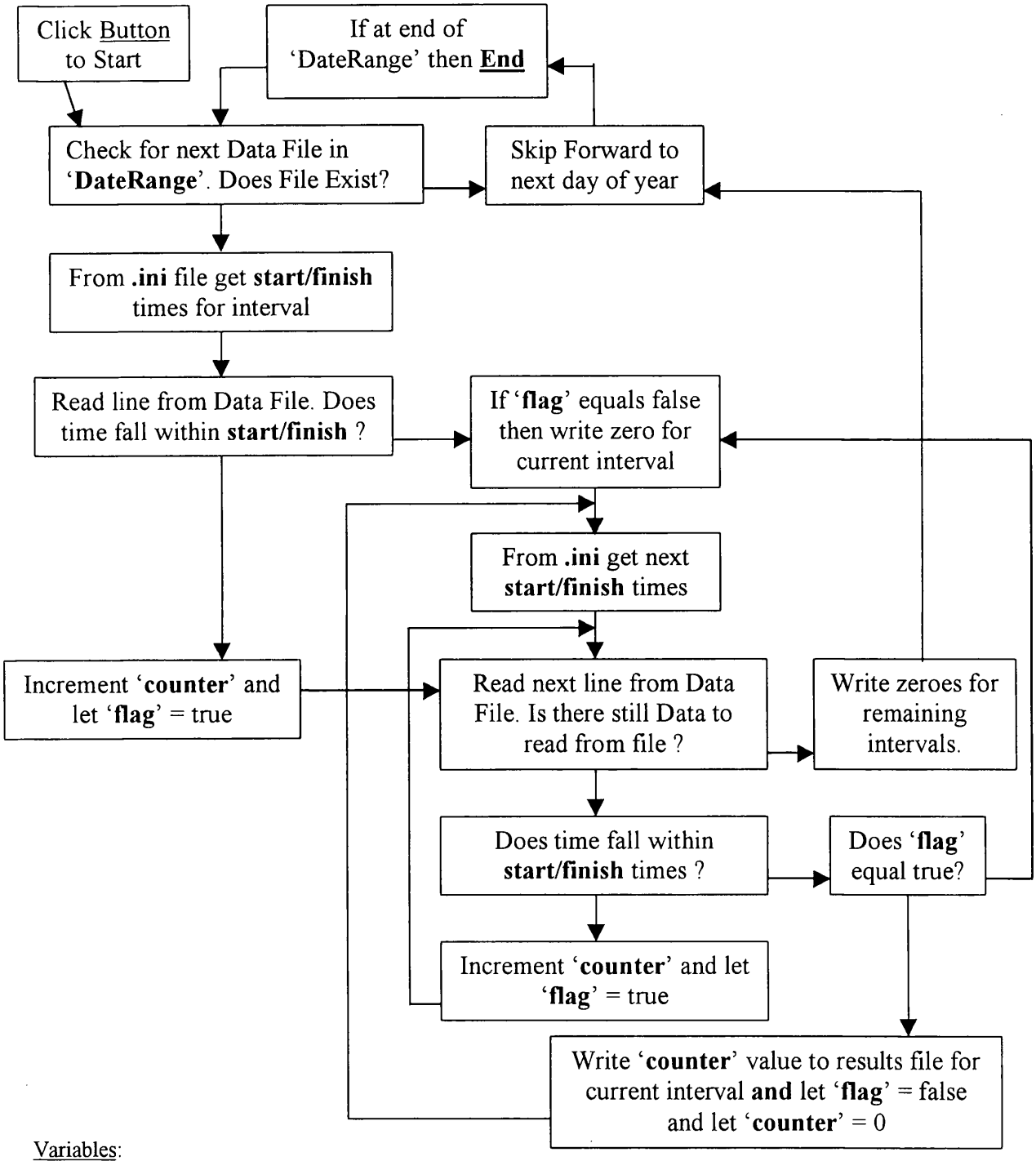


Figure 8. Screenshot of FileParse program to illustrate the graphical interface. Top-right panel contains controls for perch data parsing operations, middle-right panel contains controls for mass data parsing operations. Bottom-left panel sets date range to allow data from an entire experiment to be parsed by one button click. Bottom-right panel allows the pattern of intervals used for parsing operations to be set up.

Figure 9. Flowchart diagram showing flow of execution for the '**TransformFile-PerchData**' event handler (within the FileParse program) which transforms a series of raw perch data files into a measure of hops per interval (i.e. hop rate). The full code for this function can be found in Appendix 2, lines 480 to 613.

Note: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right represents 'no'.



Variables:

'**DateRange**' is a user defined pair of dates between which the program will automatically search for raw data files.

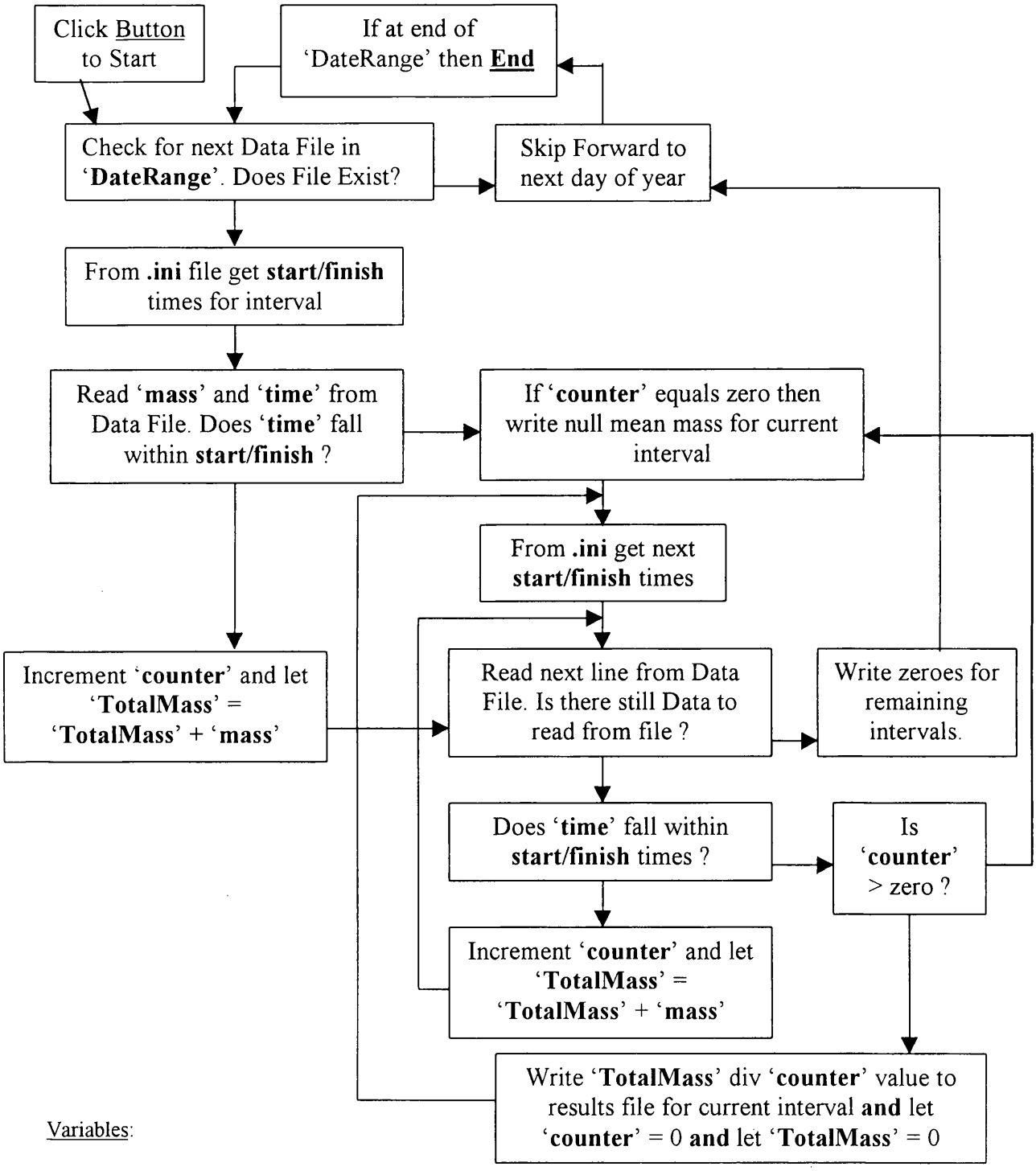
'**start/finish**' are read from the **.ini** file and define the intervals into which the day is to be subdivided (for the purposes of all analyses this is 48 half hour intervals)

'**flag**' is a Boolean variable that indicates whether any hops have fallen within the current interval as defined by the .ini file.

'**counter**' keeps a running total of hops within a given interval.

Figure 10. Flowchart diagram showing flow of execution for the '**TransformFile-MassData**' event handler (within the FileParse program) which transforms a series of raw mass data files into a measure of mean mass per interval. The full code for this function can be found in Appendix 2, lines 294 to 469.

Note: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right represents 'no'.



Variables:

'**DateRange**' is a user defined pair of dates between which the program will automatically search for raw data files.

'**start/finish**' are read from the **.ini** file and define the intervals into which the day is to be subdivided (for the purposes of all analyses this is 48 half hour intervals)

'**TotalMass**' is a running total of all the values for '**mass**' within the current interval

'**counter**' keeps a running total of the number of '**mass**' readings within a given interval and is subsequently used to calculate mean mass from '**TotalMass**'.

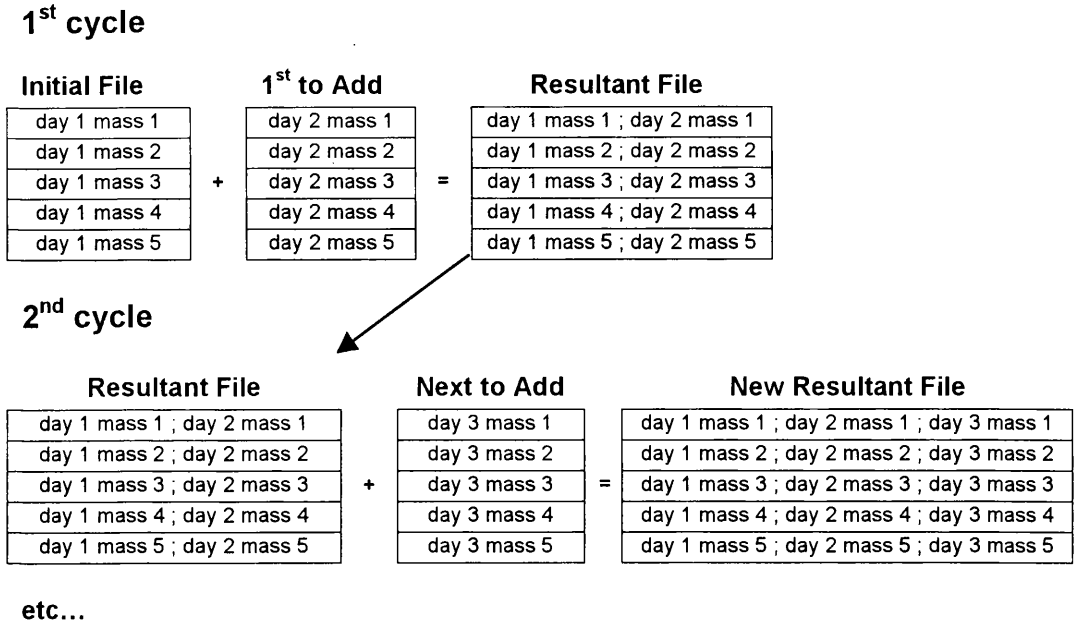
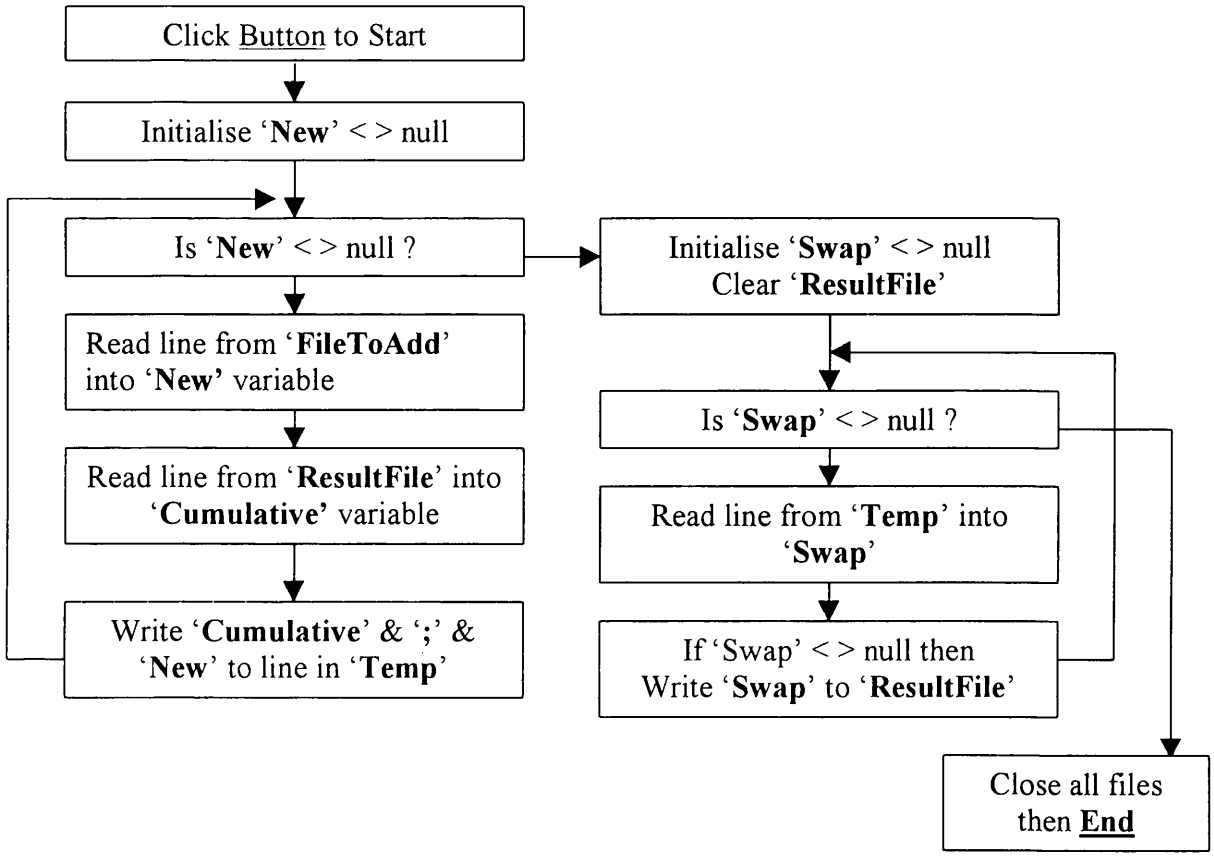


Figure 11. Diagram indicating how '*CombineFiles*' routine sequentially combined individual textfiles into one cumulative, semicolon-delimited file. Each individual table represents a single textfile. Any number of equal length textfiles could be combined in this manner.

Figure 12. Flowchart diagram showing flow of execution for the '**CombineFiles**' event handler (within the FileParse program) which allows unlimited textfiles of equal length to be added to one another, side by side. The full code for this function can be found in Appendix 2, lines 965 to 1008.

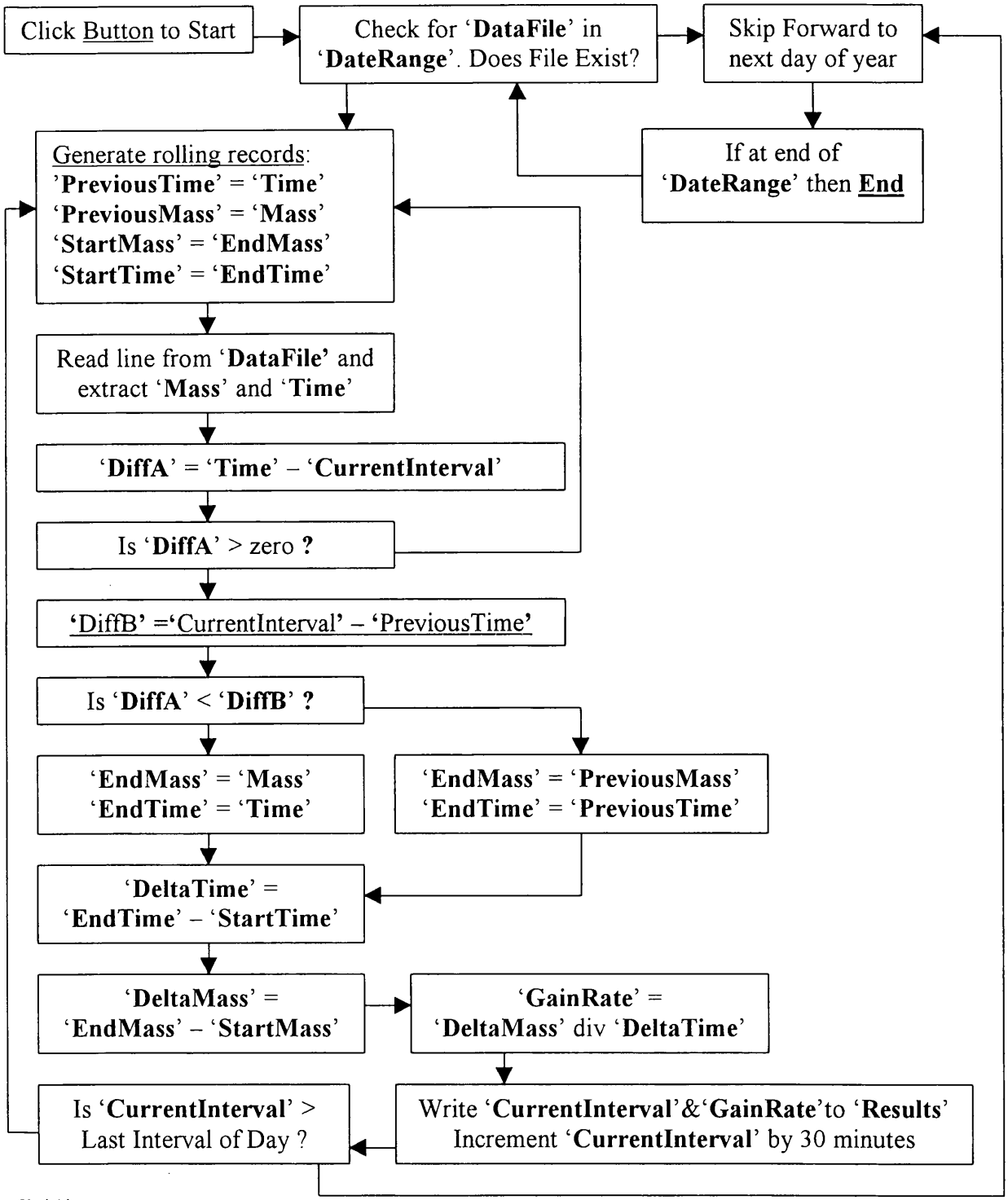
Note: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right represents 'no'.



Variables:

'**New**' is a variable into which each line of '**FileToAdd**' is sequentially read. When '**New**' equals null this signifies that the end of the file has been reached (i.e. no more data)
'**Cumulative**' is a variable in which each line of '**ResultFile**' is sequentially read.
'**FileToAdd**' is the user selected textfile which is to be added to the existing textfile named '**ResultFile**'.
'**Temp**' is a third textfile used as a temporary read/write location.
'**Swap**' is a variable into which lines from '**Temp**' are sequentially read prior to being written over into '**ResultFile**'. When '**Swap**' equals null this signifies that the end of the file has been reached.

Figure 13. Flowchart diagram showing flow of execution for the '**MassGainPerInterval**' event handler (within the FileParse program) which transforms a series of raw mass data files into a measure of rate of mass gain per half hour interval. The full code for this function can be found in Appendix 2, lines 1196 to 1328. **Note**: where there are two possible pathways, a down arrow represents 'yes' and an arrow to the right/left represents 'no'.



Variables:
'**DateRange**' is a user defined pair of dates between which the program will automatically search for raw data files.
'**Mass**' and '**Time**' are values read from current line of textfile being parsed
'**EndTime**' is the value from the file being parsed which is nearest to end of '**CurrentInterval**'. '**EndMass**' is the associated '**Mass**' reading.
'**PreviousMass**' and '**PreviousTime**' are '**Mass**' and '**Time**' values from penultimate line of textfile
'**CurrentInterval**' denotes the half hour period for which a '**GainRate**' is sought
'**DiffA**' and '**DiffB**' are used to determine whether the time of the '**Mass**' reading immediately before '**CurrentInterval**' or that immediately after is closest. '**Results**' is the resultant textfile containing the Mass Gain Rate values.

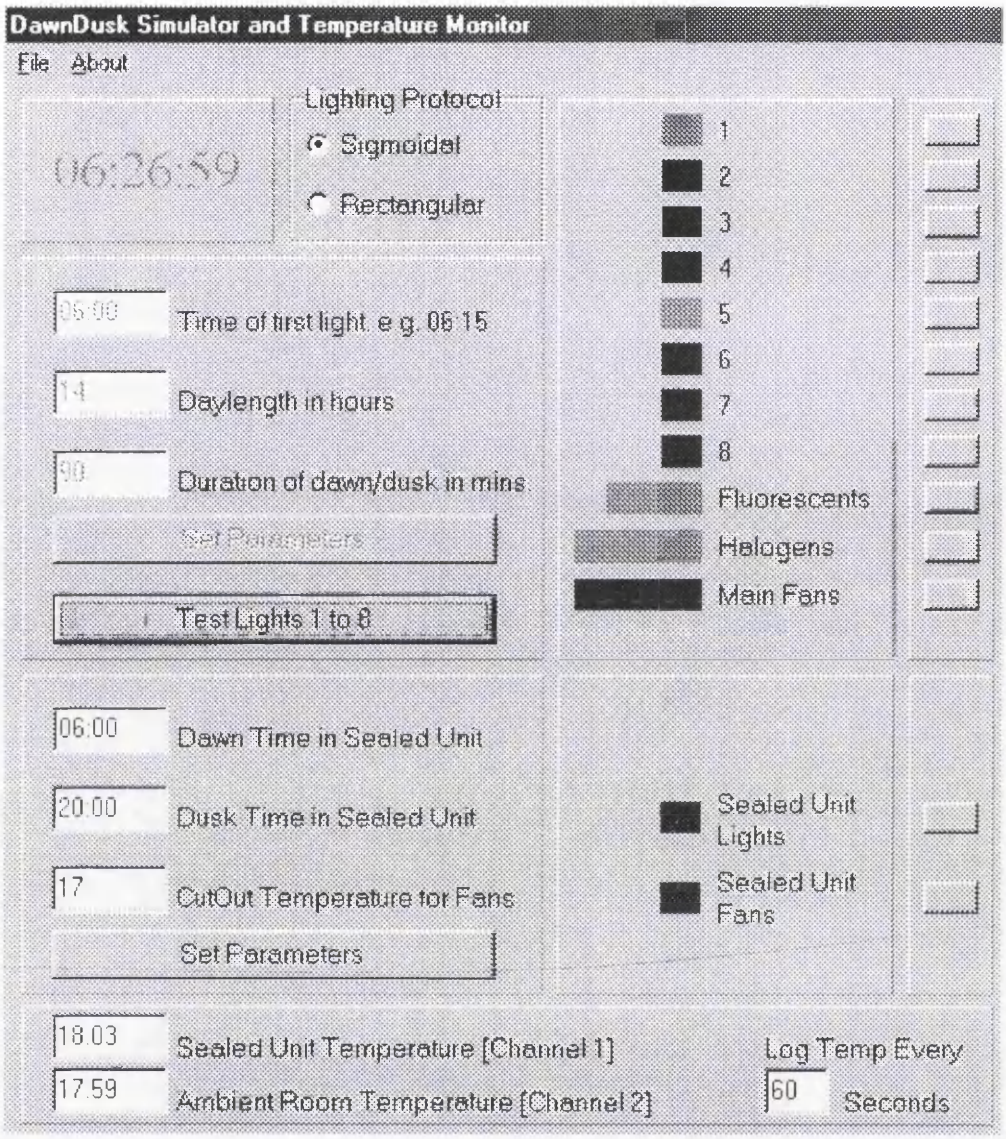


Figure 14. Screenshot of DawnDusk program to illustrate the graphical interface. Shows input boxes for setting times and temperature (left hand side), and status-indicators and manual override/test buttons (right hand side).

Chapter 3. Sigmoidal versus rectangular lighting regimes: effects on diel patterns of body mass and display hopping in male zebra finches

Abstract

A large part of the work performed using captive zebra finches has taken place in artificially lit indoor aviaries where room lights had only two states; on or off (rectangular). Such arrangements deny individuals of any extrinsic cues indicating the imminent onset of light or darkness. These cues will likely have roles in the timing and duration of courtship routines and foraging patterns associated with dawn and dusk. This study used a novel lighting apparatus that caused light intensity gradually to increase at dawn and decrease at dusk (sigmoidal) in an attempt to mimic a more natural condition. Male zebra finches were presented with a female stimulus bird and body masses and hopping activity were compared as lighting regime was switched from sigmoidal to rectangular then back to sigmoidal. There was no significant effect on overall body mass, but there was a significant decrease in body mass following the transition from the rectangular to the second sigmoidal lighting regime, and possible reasons for this are discussed. There was evidence of nonsignificant changes in mass trajectories, with a tendency for birds to act more cautiously insofar as target masses were attained sooner under the rectangular regime. There were no significant effects of lighting regime on courtship hopping activity, and hopping rate trajectories were consistent between treatments. However, in the cases of both mass regulation and hopping activity there was evidence that individuals were able to anticipate the onset of darkness under the rectangular regime which indicated the presence of an intrinsic mechanism, probably a circadian clock. Despite individuals being able to anticipate sudden dark/light changes we feel that experimenters should consider adopting more naturalistic lighting protocols.

Introduction

A vast number of experiments have been performed on caged birds, particularly zebra finches, in the laboratory environment (e.g. see Zann, 1996). Some of these studies have used natural daylight to illuminate the laboratory (e.g. Meijer et al., 1996; Pravosudov & Grubb, 1997), others have used a combination of natural and artificial light (e.g. Veasey et al., 1998) and in a majority of cases only artificial lighting was employed or the lighting conditions were not specified (e.g. Adret, 1993; Balzer & Williams, 1998; Collins, 1995).

Much recent work has highlighted the importance of a full natural spectrum (including a UV component) in avian visual systems (for a review see Tovée, 1995). Colour rendition in artificially lit settings can be made more naturalistic by employing special incandescent or fluorescent lighting devices (e.g. Bennett et al., 1996; Cuthill et al., 1997). However, the best and most cost-effective means of providing true colour perception and perception of time relative to light intensity changes is to use naturally lit aviaries. Natural daylight it is not always used since it prevents operators from controlling other aspects of the birds' environment.

For instance, experimenters may wish to have precise control over photoperiod or ambient temperature and be able to manipulate these factors (Ekman & Hake, 1990; Meijer et al., 1996; Weathers & Greene, 1998). Also it may be desirable to exclude sensory stimulation arising from outside of the birds' cages. While outdoor aviaries allow birds to perceive the full natural spectrum they also expose them to the prevailing ambient temperature, weather, ambient olfactory and auditory signals and possible provocation by wild or domestic animals outwith their cages, any of which may significantly affect behaviour or state (see Lilliendahl, 1997, 1998). In this event most experimenters will choose to house their birds in indoor aviaries where photoperiod, temperature and background noise can be predictably controlled. Photoperiod has traditionally been controlled by some form of electro-mechanical or electronic time switch which causes room lights to go on and off in unison at programmed times. The consequences of this are that the light intensity changes abruptly from zero to full at dawn and full to zero at dusk, with no intermediate levels. Clearly such a condition is

extremely unnatural and denies subjects of any extrinsic cues pertaining to the imminent onset of full light or darkness.

For experiments concerned with sexual selection and reproduction in birds, dawn and dusk are key times of day when courtship and copulation rates tend to be highest and behaviour generally most elaborate (e.g. Kacelnik & Krebs, 1982; Houston & McNamara, 1987; Cuthill & MacDonald, 1990; Andersson, 1994; Thomas 1999a & 1999b). Changing light intensities will undoubtedly have roles in the timing and duration of courtship routines associated with dawn and dusk and also in temporal foraging patterns. It is difficult to imagine that their absence will not alter the behavioural repertoire of caged birds.

While issues surrounding the quality of artificial lighting, particularly concerning UV spectra, have been investigated in recent years it appears that the importance of naturalistic light intensity changes has been overlooked. This experiment was designed to test the effectiveness of a novel lighting apparatus that attempted to mimic the changing light intensities associated with dawn and dusk. It also allowed an insight into the relative importance of extrinsic and intrinsic time-of-day cues in regulating body mass and courtship hopping behaviours. We presented male zebra finches with a female stimulus bird and compared their mass regulation and courtship behaviours, which are typically most labile at dawn and dusk, under an all-or-nothing (rectangular) lighting regime as opposed to one where light intensity was allowed to increase and decrease gradually at dawn and dusk respectively (sigmoidal).

Methods

We performed this experiment using a group of eight male and a separate group of eight female zebra finches. The two groups had no prior experience of each other. All birds were sexually mature and wore a single numbered orange leg band. The sexes were housed in visual but not auditory isolation prior to the experiment. Room temperature was kept at approximately 18°C by means of thermostatically controlled heaters. Birds were provided with *ad libitum* water and food in the form of a proprietary seed mix and in addition were provided with shell grit and cuttlebone.

Eight experimental cages (1.5 x 0.3 x 0.4 m high, figure 1) were employed, each consisting of three equally sized sections: food, courtship and stimulus. During treatments the experimental male had access to the food and courtship sections and the stimulus section was occupied by a female zebra finch. The food and courtship sections were separated by opaque baffles which permitted the male to move freely from one section to the other but prevented him from simultaneously feeding and being in visual contact with the female. The courtship and stimulus sections were separated by a transparent Perspex panel (0.15 cm thick). The food section contained a food hopper and water dish together with shell grit and cuttlebone. The courtship section contained a set of four microswitch perches that recorded the temporal and spatial patterns of hopping activity (Swaddle & Cuthill, 1994). The four perches, made from 5 cm lengths of wooden dowel, were arranged in a square with opposite pairs set 12 cm apart and each perch 14 cm above the cage floor. The set of hop perches was itself mounted on top of an electronic balance (Sartorius PT-610) reading to 0.1 g accuracy. The eight balances were connected to a PC that recorded real-time changes in the body masses of the male birds, and simultaneously corrected for drift from zero in the balances' outputs (see Chapter 2 for further details).

The stimulus section was equipped with two perches set at the same height as the microswitch perches. The wall of the stimulus section against which female stimulus birds were viewed was white. The stimulus section was bedded with composted wood bark but the food and courtship sections were not bedded in order to prevent excessive debris being deposited by the males on either the perch mechanisms or the balances.

The prevailing photoperiod was 14:10 L:D provided by a ceiling mounted linear array of eight 40 watt incandescent daylight bulbs and six ceiling mounted 58 watt fluorescent tubes with diffusers. Daylight was completely excluded from the laboratory. Lighting was controlled by computer via an expansion card (PC 263, Amplicon Liveline, UK) and was applied either on an all-or-nothing basis (rectangular regime) or through sequential switching of the individual lighting elements (sigmoidal regime). Under the sigmoidal regime there were nine evenly spaced switching events over a 90 minute period, which at dawn switched on the eight incandescent bulbs in sequence followed by the fluorescent tubes as a unit, and at dusk acted in the reverse order. This produced an approximately exponential change in lighting intensity intended to provide similar cues to a natural daylight dawn and dusk (see figure 2). The time elapsed from the first light going on at dawn to the last going off at dusk was 14h.

The experiment consisted of three treatments which involved exposing birds first to a sigmoidal lighting regime (treatment 1) followed by a rectangular regime (treatment 2) then by a second sigmoidal regime identical to the first (treatment 3). Each treatment lasted twelve days and data from only the final seven days of each were used in the analyses, giving in effect a five day acclimation period for each changeover in lighting regime. For four weeks prior to the first treatment the lighting regime was sigmoidal with the same parameters as treatments 1 and 3.

We investigated the effect of lighting regime on within-individual variation in body mass and hop rate using repeated measures ANOVA designs in SPSS 7.5.1. In addition we sought to investigate the effect of treatment on the diurnal mass trajectory and organisation of hopping behaviour by considering the interaction between treatment and time of day (treatment*time). The data for each individual were averaged across the seven days of each treatment. Body mass and display hopping data were condensed into mean mass per half hour interval and total number of hops per half hour (i.e. hop rate) over the course of the daylight period of each day. The model therefore, specified two factors: treatment (three levels) and time of day (twenty-eight levels). The identity of individual males was inherent in the design of the model.

We also looked specifically at contrasts between treatments to test whether the transition from sigmoidal to rectangular lighting regime, or the transition from rectangular back to sigmoidal had any effect on body mass regulation or hopping activity. In addition we tested the contrast between the two sigmoidal treatments to determine if behaviour remained consistent between them.

We present univariate repeated measures ANOVA statistics and where assumptions about sphericity are violated the Huynh-Feldt epilson corrected statistics are quoted (Zar, 1996).

Results

There was no significant effect of treatment on body mass ($F_{1.12,7.85} = 1.388$, $P = 0.28$), nor was there a significant effect of treatment on hopping activity ($F_{1.25,8.78} = 0.477$, $P = 0.55$). The non-significant interactions between treatment and time of day showed that body mass trajectories (Figure 3; $F_{10.24,71.67} = 1.570$, $P = 0.13$) were consistent between light regimes, as were the hopping activity trajectories (Figure 4; $F_{6.51,45.56} = 1.035$, $P = 0.42$).

Contrasts revealed that there were no significant differences between sigmoidal treatments (1 & 3) for either body mass ($F_{1,7} = 0.985$, $P = 0.35$) or for hopping activity ($F_{1,7} = 0.037$, $P = 0.85$). Similarly, there were no significant differences between the first sigmoidal treatment and the rectangular treatment (i.e. treatments 1 & 2) for either body mass ($F_{1,7} = 0.095$, $P = 0.77$) or hopping activity ($F_{1,7} = 0.503$, $P = 0.50$). However, there was a significant difference between the rectangular treatment and the second sigmoidal treatment (treatments 2 & 3) for body mass ($F_{1,7} = 31.39$, $P = 0.001$), with mass being consistently lower in the sigmoidal treatment by between 1.1% and 3.5% (of the masses during the treatment 2) across the course of the day. There was no significant trend for hopping activity to differ between the treatments ($F_{1,7} = 3.029$, $P = 0.125$).

Discussion

The results showed a significant difference in body mass between the rectangular (second) treatment and the ensuing sigmoidal (third) treatment, with body masses during the final sigmoidal treatment being lower by between 1.1% to 3.5% across the day. There was the possibility that this difference was due to a systematic factor or factors, for example a change in room temperature (c.f. Lehikoinen, 1987; Meijer et al., 1996). However, the absence of any corresponding difference between treatments 1 and 2 makes this less likely. Also, although room temperature was not monitored during this particular experiment, temperature data from four later experiments in the same laboratory indicated that temporal fluctuations in temperature were minimal owing to the use of thermostatically controlled heating and ventilation.

Following the rectangular treatment, the males had five days to re-acclimatise to the new lighting conditions of the second sigmoidal treatment, whereas before the first sigmoidal treatment, the males had the five experimental acclimation days in addition to the 28 days prior to the experiment during which the lighting conditions had also been sigmoidal. Passerines are known not to feed when light levels fall below a certain level (Kacelnik, 1979). Therefore, during the sigmoidal treatments, the periods nearest to dawn and dusk, when light intensities were lowest, may have shortened the perceived length of the feeding day. In the rectangular treatment however, feeding would have been possible for the full fourteen hours from the lights turning on, until they turned off again. Some studies have indicated that shorter daylengths caused the body masses of small passerines to increase (e.g. Lehikoinen, 1987; Meijer et al., 1994), however, it was recently demonstrated that zebra finches decreased their body masses in response to shortening daylength, and vice versa (Meijer et al., 1996). The observed decrease in body mass between treatments 2 and 3 may therefore have been a reaction to a perceived shortening of the feeding day length following the transition from the rectangular to the sigmoidal lighting regime. During the 28 acclimation days prior to the experiment, males may have adapted to the prevailing sigmoidal lighting conditions and learned to utilise the periods of lower light around dawn and dusk more fully. The effect of this would have been that following the first transition, from sigmoidal to rectangular lighting, the change in perceived feeding day would have been relatively smaller, and

therefore the change in mass (in this case an expected increase) would have been smaller too.

Although there were no significant differences between mass trajectories or hopping activity levels, there was a slight tendency for the rate of mass gain to be higher and hopping activity to be lower during the initial hour of the rectangular treatment. One explanation for the dawn chorus in passerines relates to light intensity. Kacelnik (1979) proposed that foraging may be less profitable at dawn due to poor vision as a consequence of low light levels. In this scenario it may be more advantageous for a male bird to engage in singing behaviour (which is analogous to courtship hopping) and postpone feeding until light level increases. It may be that the tendency for an increased rate of body mass gain and decreased display effort during 'dawn' of the rectangular treatment relates to Kacelnik's hypothesis. The instant onset of full light intensity under the rectangular treatment as opposed to the sigmoidal treatments may have stimulated an earlier and/or increased foraging effort which was traded off against a slightly reduced display effort.

Although there were no significant differences in body mass trajectory (treatment*time interaction) figure 3 suggested a nonsignificant tendency for the trajectory under the rectangular treatment to differ from the other two. Under the rectangular treatment there appeared to be a tendency for the birds to adopt a more guarded approach towards mass regulation insofar as they adopted a greater rate of mass gain during the first three hours of the day. Then, just before midday the mass trajectory flattened briefly before resuming a near linear rate of increase up until one hour before darkness prevailed. In contrast during the first sigmoidal treatment there was a more typical mass trajectory (chapters 4, 5 & 6 of this thesis; McNamara et al., 1987) with a smoother and more linear rate of increase from dawn until midday, followed by a slight plateau typical of midday, and then a slow increase until approximately three hours before dusk when the rate of mass gain increased sharply towards the dusk target mass.

It was interesting that under the rectangular treatment where there were no external cues to suggest darkness was imminent, the rate of body mass gain became negative during the last hour of light, similar to the situation under both sigmoidal treatments. This effect may have been due to a physiological mechanism acting on body mass whereby

once a maximum body mass is attained a feedback loop kicks in to maintain and/or reduce it. Another more likely possibility is that individuals were able to anticipate darkness by means of a circadian rhythm that coincided with the prevailing photoperiod. The presence of circadian rhythms is widely acknowledged throughout the animal kingdom (Aschoff, 1981; Chadwick & Ackrill, 1995). Distinct circadian rhythms driven by periodic secretion of melatonin linked to the light cycle have been demonstrated in passerines (Hau & Gwinner, 1992; Hau & Gwinner, 1994; Hau & Gwinner, 1996; Hau & Gwinner, 1997) and specifically in zebra finches (Vant Hof & Gwinner, 1999). The circadian clock may have allowed birds in the present study to anticipate darkness irrespective of extrinsic cues such as gradually fading light intensity, as it would have become entrained to the prevailing daylength which was constant before and during the experiment

There was no effect of treatment on directed hopping activity, which implied that overall levels of activity were not affected by the different lighting regimes. Nor was there any effect of the treatment*time interaction on directed hopping activity and it is clear from figure 4 that the trajectories under all three treatments were very similar. Again it was apparent that under the rectangular regime the males anticipated the onset of darkness and increased their rate of hopping activity in the manner typical of dusk. The point during the rectangular treatment at which hopping activity began to increase towards its dusk peak was approximately thirty minutes later than during the sigmoidal treatments but the rate of increase and ultimate level were very similar between treatments. It was difficult to imagine how this increased hopping activity in anticipation of darkness could be mediated by any feedback mechanism, in contrast to the case of body mass regulation which may have been self-limiting, and therefore the favoured explanation is again that involving an intrinsic clock.

Zebra finches are the most widely used passerine for captive experimental work (Zann, 1996). Given the controllability and constancy of indoor versus outdoor aviaries it is likely that the majority of experimenters will continue to favour indoor housing. There will always be restrictions on the degree to which experimenters can seek to mimic natural conditions, and in any event a human perspective as to what constitutes ideal may vary considerably from that of the animal subject (e.g. Bennett et al., 1996). Common sense dictates that the all-or-nothing lighting arrangements in place in many

laboratories differ dramatically from a natural daylight cycle. This study has demonstrated that adopting a light intensity regime that tended towards natural gradients had certain significant effects on body mass regulation strategies. Given these effects and the relative simplicity of the lighting apparatus used, we believe that more experimenters should consider adopting similar approaches to lighting protocols in order to allow the possible benefits to be evaluated further.

References

Adret, P. 1993. Operant conditioning, song learning and imprinting to taped song in the zebra finch. Animal Behaviour, 46, 149-159.

Andersson, M. 1994. Sexual Selection. Princeton University Press, Princeton.

Aschoff, J. 1981. Biological clocks in birds. Proceedings of the XVII Congress of International Ornithology: 113-136.

Balzer, A. L. & Williams, T. D. 1998. Do female zebra finches vary primary reproductive effort in relation to mate attractiveness? Behaviour, 135, 297-309.

Bennett, A. T. D., Cuthill, I. C., Partridge, J. C. & Maier, E. J. 1996. Ultraviolet vision and mate choice in zebra finches. Nature, 380, 433-434.

Chadwick, D. J. & Ackrill, K. (eds) 1995. Circadian Clocks and their Adjustment. Chichester, England: John Wiley & Sons Ltd.

Collins, S. A. 1995. The effect of recent experience on female choice in zebra finches. Animal Behaviour, 49, 479-486.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Cuthill, I. C., Hunt, S., Cleary, C. & Clark, C. 1997. Colour bands, dominance, and body mass regulation in male zebra finches (Taeniopygia guttata). Proceedings of the Royal Society of London Series B, 264, 1093-1099.

Ekman, J. B. & Hake, M. K. 1990. Monitoring starvation risk: adjustments of body reserves in greenfinches (Carduelis chloris L.) during periods of unpredictable foraging success. Behavioral Ecology, 1, 62-67.

Fraser, N. H. C. & Metcalfe, N. B. 1997. The costs of becoming nocturnal: feeding efficiency in relation to light intensity in juvenile Atlantic Salmon. Functional Ecology, 11, 385-391.

Hau, M. & Gwinner, E. 1992. Circadian entrainment by feeding cycles in house sparrows, Passer domesticus. Journal of Comparative A, 170, 403-409.

Hau, M. & Gwinner, E. 1994. Melatonin facilitates synchronization of sparrow circadian rhythms to light. Journal of Comparative Physiology A, 175, 343-347.

Hau, M. & Gwinner, E. 1996. Food as a circadian zeitgeber for house sparrows: The effect of different food access durations. Journal of Biological Rhythms, 11, 196-207.

Hau, M. & Gwinner, E. 1997. Adjustment of house sparrow circadian rhythms to a simultaneously applied light and food zeitgeber, Physiology & Behavior, 62, 973-981.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

Kacelnik, A. 1979. Studies on foraging behaviour and time budgeting in the great tit (Parus major). Unpublished D Phil Thesis, University of Oxford.

Lehikoinen, E. 1987. Seasonality of the daily weight cycle in wintering passerines and its consequences. Ornis Scandinavica, 18, 216-226.

Lilliendahl, K. 1997. The effect of predator presence on body mass in captive greenfinches. Animal Behaviour, 53, 75-81.

Lilliendahl, K. 1998. Yellowhammers get fatter in the presence of a predator. Animal Behaviour, 55, 1335-1340.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioral Ecology and Sociobiology, 20, 399-405.

Meijer, T., Möhring, F. J. & Trillmich, F. 1994. Annual and daily variation in body mass and fat of starlings Sturnus vulgaris. Journal of Avian Biology, 25, 98-104.

Meijer, T., Rozman, J., Schulte, M. & Stach-Dreesmann, C. 1996. New findings in body mass regulation in zebra finches (Taeniopygia guttata) in response to photoperiod and temperature. Journal of Zoology, 240, 717-734.

Pravosudov, V. V. & Grubb, T. C. 1998. Management of fat reserves in tufted titmice (Parus bicolor): evidence against a tradeoff with food hoards. Behavioural Ecology and Sociobiology, 42, 57-62.

Thomas, R. J. 1999a. The effect of variability in the food supply on the daily singing routines of European robins: a test of a stochastic dynamic programming model. Animal Behaviour, 57, 365-369.

Thomas, R. J. 1999b. Two tests of a stochastic dynamic programming model of daily singing routines in birds. Animal Behaviour, 57, 277-284.

Tovée, M. J. 1995. Ultra-violet photoreceptors in the animal kingdom: their distribution and function. Trends in Ecology and Evolution, 10, 455-460.

Vant Hof, T. J. & Gwinner, E. 1999. Influence of pinealectomy and pineal stalk deflection on circadian gastrointestinal tract melatonin rhythms in zebra finches (Taeniopygia guttata). Journal of Biological Rhythms, 1999, 185-189.

Veasey, J. S., Metcalfe, N. B. & Houston, D. C. 1998. A reassessment of the effect of body mass upon flight speed and predation risk in birds. Animal Behaviour, 56, 883-889.

Weathers, W. W. & Greene, E. Thermoregulatory responses of bridled and juniper titmice to high temperature. The Condor, 1998, 100, 365-372.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Zar, J. H. 1996. Biostatistical Analyses. 3rd Edition. New Jersey: Prentice Hall International Inc.

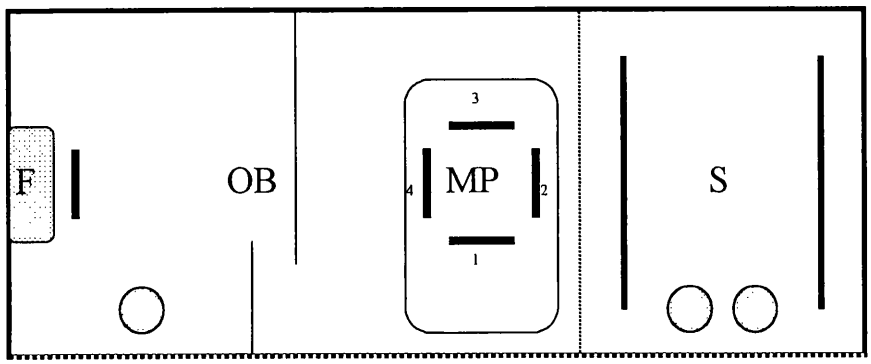


Figure 1. The experimental cage viewed from above. F = feed hopper, OB = opaque baffles, MP = microswitch perches mounted on electronic balance, S = stimulus cage. Heavy bars represent perches, dotted lines represent wire cage front and Perspex panel separating courtship and stimulus compartments. Shaded circles are food and water dishes.

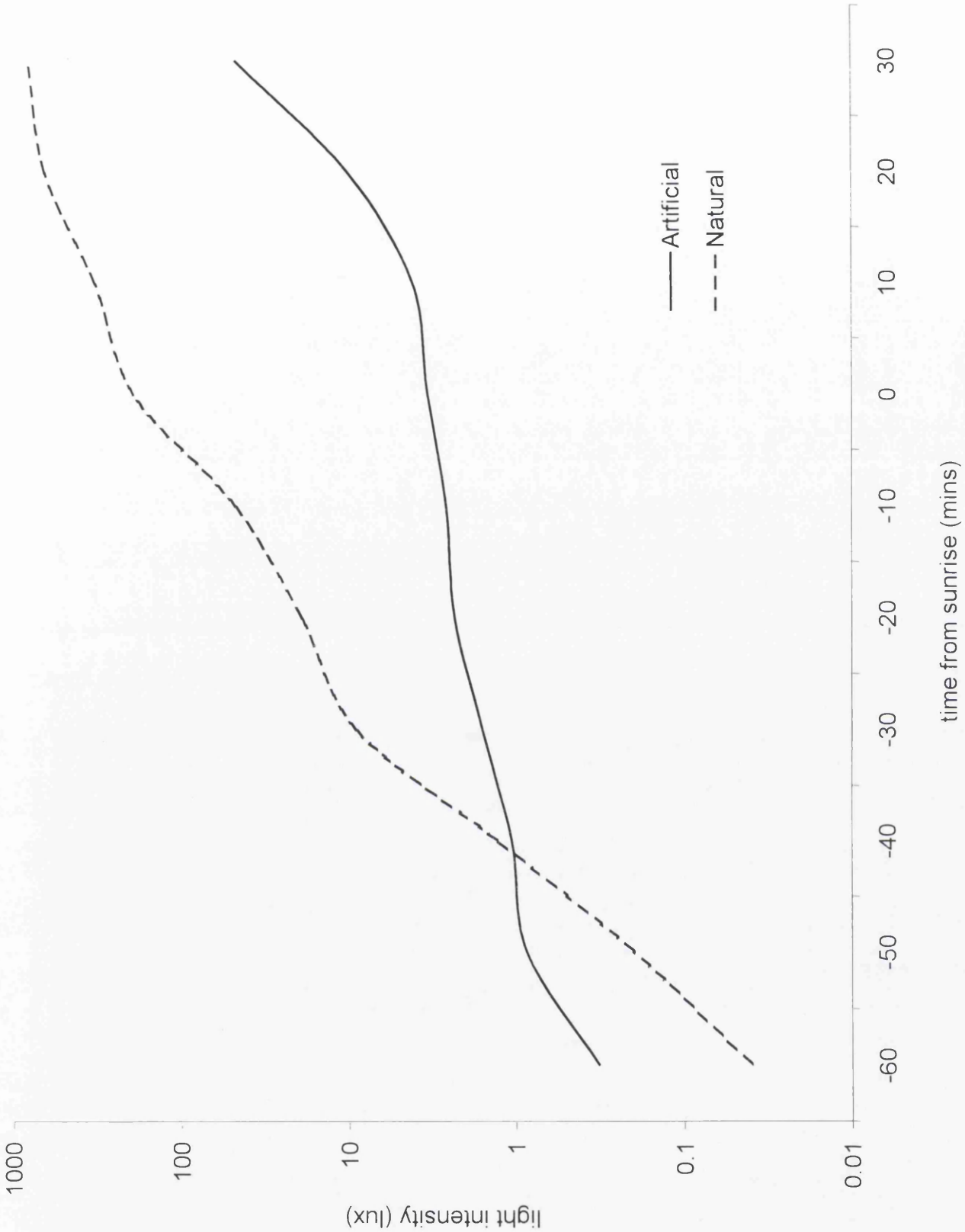


Figure 2. Relative shapes of light intensities generated by natural dawn compared to the artificial dawn simulation achieved by switching lights on in sequence. Data on natural light intensities are taken from Fraser & Metcalfe, 1997.

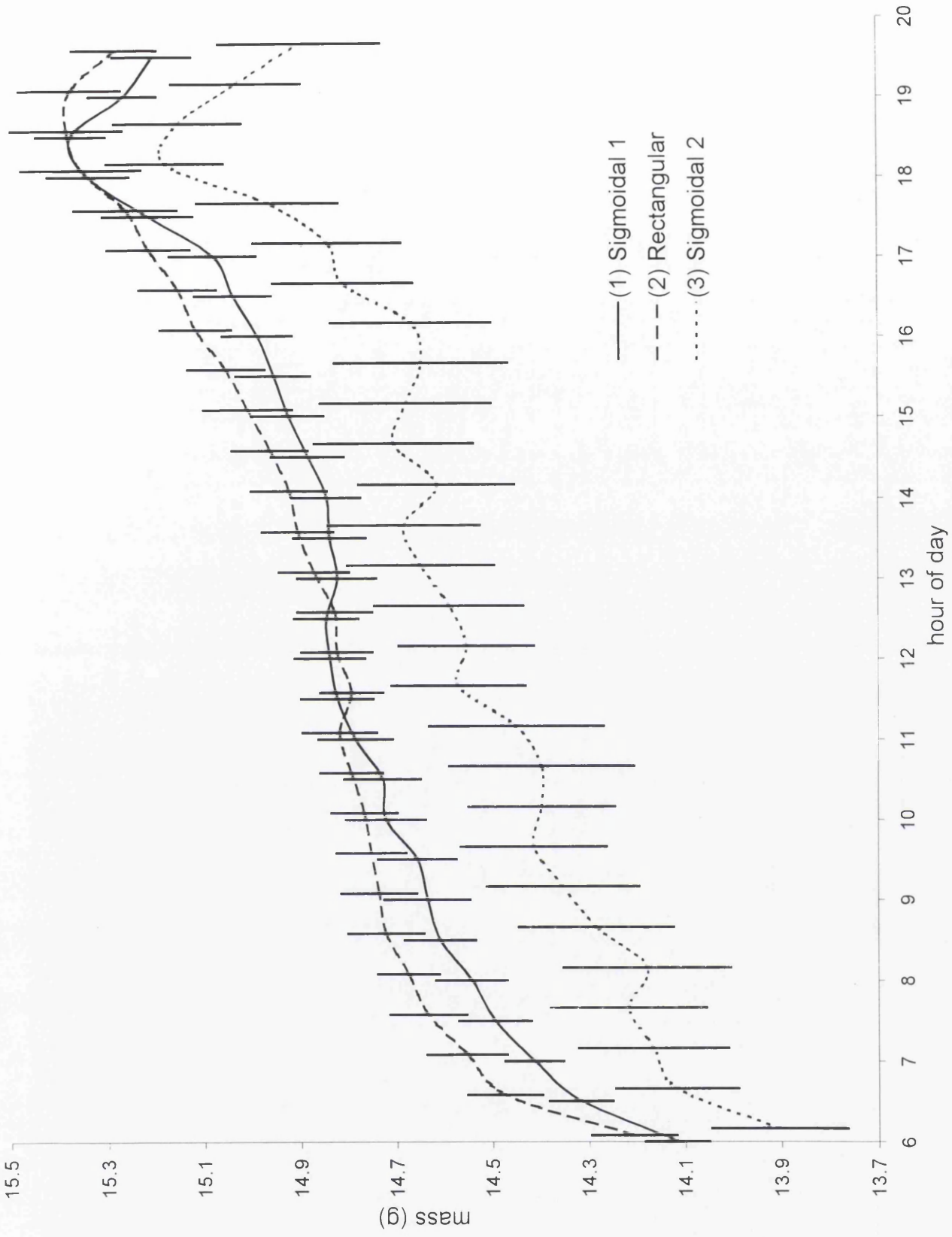


Figure 3. Mean body mass trajectories (± S.E.) of male zebra finches presented with a female during three lighting regimes. X-axis values are offset between treatments by 5 minutes for clarity (to prevent error bars from overlapping). Treatment 1, first sigmoidal; Treatment 2, rectangular; Treatment 3, second sigmoidal (presented in order 1, 2 then 3).

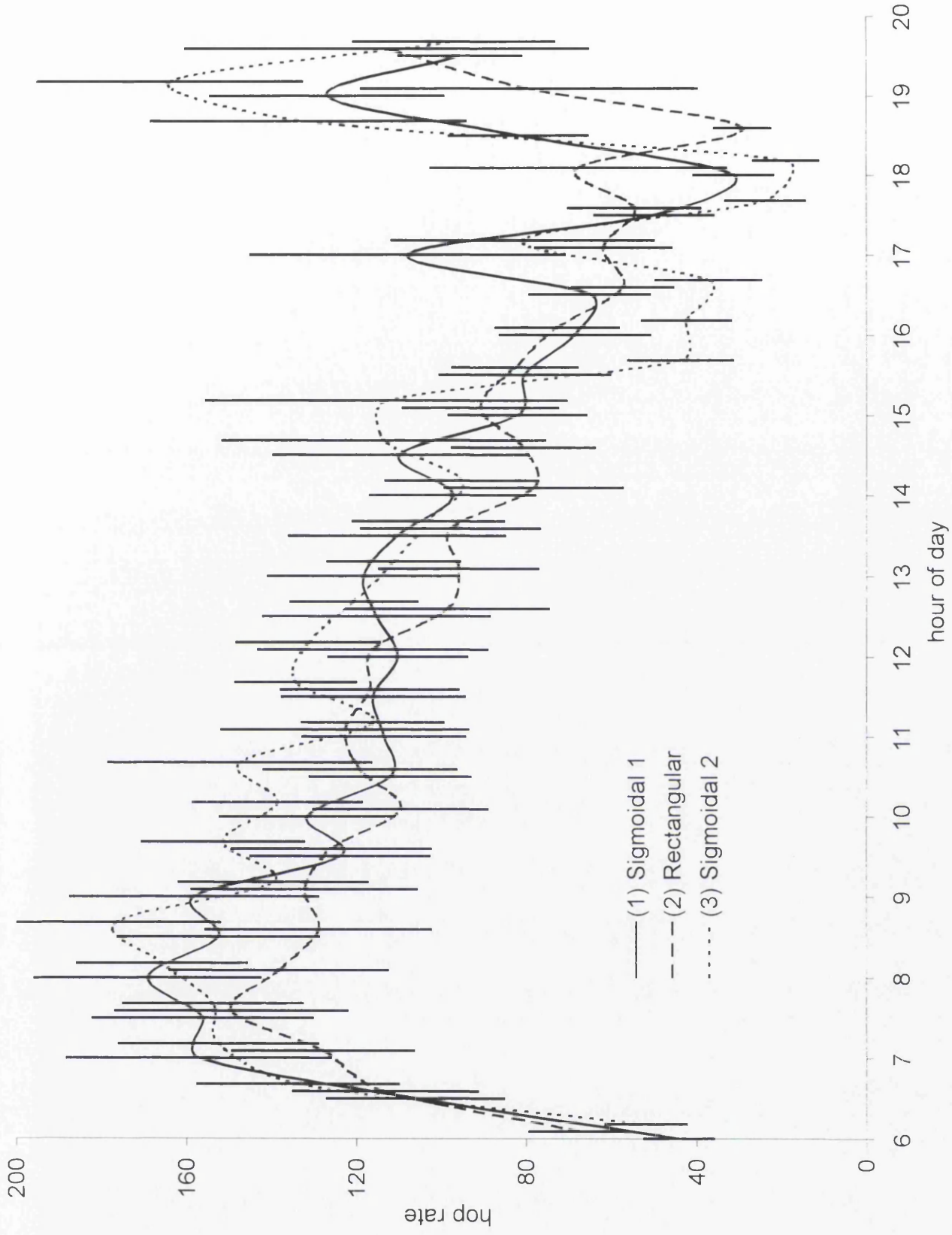


Figure 4. Mean hop rate trajectories (± S.E.) of male zebra finches. Treatments and data presented as in Figure 3.

Chapter 4. Trade-off between body mass and reproductive behaviour in male zebra finches

Abstract

Animals face potential trade-offs in resource allocation when competing activities are mutually exclusive. In particular, the trade-off in time allocated to reproduction as opposed to foraging predicts that time and energy invested in courtship will influence body mass regulation strategies. We investigated the dynamic relationship between body mass regulation and investment in reproduction (quantified in terms of directed hopping activity) in male zebra finches (Taeniopygia guttata). Males were presented with four treatments, which varied in their potential to be courted, in a repeated-measures design. The treatments were; female zebra finch, male zebra finch, female bengalese finch (Lonchura domestica) and 'no bird'. We found significant effects of treatment on both mean body mass and directed hopping activity, with lowest mean body masses and highest mean activity levels during the female zebra finch treatment. There was a significant effect of treatment on the temporal organisation (trajectory) of hopping activity over the course of a day. There was no effect of treatment on the shape of body mass trajectories; the difference among treatments was due to a complete y-axis shifting of mass curves. The apparent anomaly of consistent mass trajectories coupled with significant variation in directed hopping activity may be explained either by strategic compensatory foraging, reduced metabolic costs associated with decreased body mass, a reorganisation of time and energy budgets or some combination of these effects. This study provides direct experimental evidence of a trade-off between investment in the soma and investment in reproduction.

Introduction

No animal can devote the same unit of either time or energy to more than one given task or behaviour. Within the boundaries of a day, therefore, resource allocation decisions must be made with reference to the various conflicting demands which present themselves. Such decisions are a key tenet of life history theory (Stearns, 1989; Even & Nicolaïdis, 1993; Rogers & Smith, 1993; Guntrip et al., 1997; Kokko, 1998).

In birds, particularly in small passerines, the strategic allocation of resources is inextricably linked to changes in body mass (Hainsworth, 1978; Ekman & Hake, 1990; Hiebert, 1991; Witter & Cuthill, 1993). Investment in somatic tissue is beneficial as a buffer against the risk of starvation but there are inevitable costs associated with increased body mass (for a review see Witter & Cuthill, 1993). The existence of these costs is underlined by the fact that birds typically maintain their fat reserves below what would be physiologically possible, and that individuals with low fat levels do not necessarily exploit all opportunities to feed and hence lay down fat (King & Farner, 1966; Blem, 1976; Lehikoinen, 1986; Ekman & Hake, 1990). The costs of increased body mass have been thoroughly documented in the literature and play a key role in phenomena such as mass-dependent predation risk (Bednekoff & Houston, 1994; Witter et al., 1994; Gosler et al., 1995; Metcalfe & Ure 1995; Bednekoff, 1996; Cresswell, 1998; Veasey et al., 1998), state-dependent foraging (Metcalfe et al., 1998, on salmon), mass-dependent risk of injury and metabolic costs (e.g. Pennycuick, 1990).

Theoretical models have been constructed to examine the optimal diurnal patterns of mate attracting behaviour and mass regulation in birds (Houston & McNamara 1987; McNamara et al., 1987; Hutchinson et al., 1993; Bednekoff & Krebs, 1995). Although these models have been constructed around mate attracting behaviour in the form of song they can equally be applied to any courtship or display behaviour provided that the behaviour in question is energetically costly and incompatible with foraging behaviour (Hutchinson et al., 1993). These criteria are met by the display repertoire of male zebra finches. Previous empirical work (Mace, 1987; Mace, 1989; Cuthill & MacDonald, 1990) has focussed only on the temporal pattern of male display behaviour, whereas it is the dynamic interaction between courtship and mass regulation that is central to the theoretical predictions. Since courtship activity tends to be greatest in the early morning

and late evening, an increase in courtship (besides being energetically expensive) may affect both the total amount of time devoted to feeding and the temporal pattern of foraging, hence altering the mean body mass and/or daily mass trajectory.

Male zebra finches (Taeniopygia guttata) follow a stereotyped pattern of courtship (Morris, 1954; Zann, 1996) which is composed mainly of hopping behaviour during which the male bird engages in serial lateral presentations (Workman & Andrew, 1986), and also of directed song in the presence of the female (Dunn & Zann, 1996a, 1996b). Engaging in mate attraction behaviour should influence the amount of time and energy invested in courtship activities, therefore theory predicts that manipulating the opportunity to attract a mate will influence body mass regulation strategy. We provide the first experimental demonstration of this by presenting to male zebra finches stimuli ('no bird', female bengalese finch (Lonchura domestica), male zebra finch and female zebra finch) which inherently varied in their potential to be courted, and measuring diurnal patterns of stereotypical hopping behaviour and body mass regulation. The first three treatments listed were controls which allowed us to distinguish between the effects of availability of a female conspecific, rather than effects due simply to the presence of another bird (whether conspecific or heterospecific).

Methods

Subjects and Housing Conditions

We conducted this experiment using two groups of birds; an experimental group and a stimulus group. The experimental group consisted of eight male zebra finches and the stimulus group consisted of a second group of eight male zebra finches together with eight female zebra finches and eight bengalese finches (Lonchura domestica). All birds used in the experiment were sexually mature. The experimental birds had had no prior contact with any of the stimulus birds or vice versa. Before and during the experiment all individuals were maintained on a 14:10 hour light:dark photoperiod at a temperature of mean ± S.D. = 18.10 ± 3.20°C. Simulated dawn and dusk was applied during the first and last 90 minutes of the day by means of the sequential switching of a lighting array. The light was provided by eight 40 watt incandescent bulbs in a ceiling mounted linear array, six ceiling mounted 58 watt fluorescent tubes with diffusers and two ceiling mounted 300 watt halogen floodlights (orientated so as to provide reflected light only). During dawn and dusk the light level increased and decreased approximately exponentially (see chapter 3, figure 2). We provided the birds with *ad libitum* food (commercial seed mix) and water. We also gave the birds shell-grit and cuttlebone which were replenished at weekly intervals.

Experimental Cages

Eight experimental cages (1.5 x 0.3 x 0.4 m high, Fig. 1) were employed, each consisting of three equally sized sections: food, courtship and stimulus. In each cage the experimental male had access to the food and courtship sections and the stimulus section was occupied by the stimulus bird when applicable. The food and courtship sections were separated by opaque baffles which permitted the male to move freely from one section to the other but prevented him from simultaneously feeding and being in visual contact with the stimulus section. The courtship and stimulus sections were separated by a transparent Perspex panel (0.15 cm thick). The food section contained a food hopper and water dish together with a microswitch feeding perch that recorded the time and duration of feeding bouts (provided that the bird chose to feed from this perch). The courtship section contained a set of four microswitch perches that recorded

the temporal and spatial patterns of hopping activity. The four perches, made from 5 cm lengths of wooden dowel, were arranged in a square with opposite pairs set 12 cm apart and each perch 14 cm above the cage floor. Male zebra finches hop repeatedly between adjacent perches or perform 'head-tail twists' as part of their courtship display (Morris, 1954; Zann, 1996), and so this apparatus provided a suitable and reliable method of automatically recording display activity (Swaddle & Cuthill, 1994). The perch nearest to the Perspex window which separated the middle and right-hand sections was 4 cm from it. The set of four hop perches was itself mounted on top of an electronic balance (Sartorius PT-610) reading to 0.1 g accuracy. The eight balances were connected to a PC which recorded real-time changes in the body masses of the male birds, and simultaneously corrected for drift from zero in the balances' outputs.

The stimulus section was equipped with two perches set at the same height as the microswitch perches, one of which was positioned 7 cm and the other 42 cm from the perspex panel. The wall of the right-hand subsection that faced the male's hop perches was white such that stimulus birds were always viewed against a white background. The stimulus section was bedded with composted wood bark. The food and courtship sections however had no substrate and the floor was plain hardboard. This was to prevent excessive debris being deposited by the males on either the perch mechanisms or the balances.

Experimental Procedure

Prior to the data logging phases of the experiment we moved the experimental male zebra finches into the individual experimental cages. The experiment consisted of an acclimation phase lasting three days followed by four data-logging phases each lasting seven full days. Between consecutive data-logging phases there was a non-data logging phase lasting three days. During non-data logging phases of the experiment experimental and stimulus birds were removed from the experimental cages and housed in four separate group cages (1 x 0.4 x 0.3 m high) according to species and sex with experimental and stimulus males kept apart. During these phases the experimental birds and stimulus birds were maintained in visual isolation from one another. The purpose of these phases was to minimise any carry-over effects of previous stimuli on the experimental males' reaction to future ones. Also, since zebra finches are naturally

gregarious birds, it was felt that this tactile contact with conspecifics would minimise any effects due to the relative isolation experienced by the experimental males during the logging phases.

Each experimental male was exposed to a different stimulus type during each data-logging phase. The four stimuli were: no stimulus bird, female bengalese finch, male zebra finch and female zebra finch. The order in which these stimuli appeared was randomised both temporally and spatially and each experimental male experienced a particular stimulus treatment once only. Experimental males and stimuli were introduced to the appropriate compartments of the experimental cages on the evening before the beginning of a treatment and removed from the experimental cages on the morning following the end of a treatment such that the period of data logging was seven full days in length.

Statistical Analyses

We investigated the effects of stimuli on within-individual variation in body mass and hop rate using repeated measures ANOVA designs in SPSS 7.5.1 (SPSS 1988). In addition we investigated the effect of treatment on the diurnal mass trajectory and organisation of hopping behaviour by considering the interaction between treatment and time of day (treatment*time). For the purposes of the analyses the data for each individual were averaged across all seven days of each treatment. Body mass and display hopping data were condensed into mean mass per half hour interval and total number of hops per half hour (i.e. hop rate) over the course of each day. The model therefore, specified two factors; treatment (four levels) and time of day (twenty-eight levels). The identity of individual males was inherent in the design of the model.

Specific contrasts between treatments were investigated to compare the effect of a female conspecific versus the other three treatments, and the effect of female conspecific versus no stimulus bird, for both body mass and hopping activity.

The repeated measures ANOVA statistics are either univariate or multivariate depending on the outcome of Mauchly's test of sphericity (Zar, 1996). Two-tailed tests of probability are used throughout.

Results

There was a significant effect of stimulus treatment on within-individual changes in body mass (treatment, $F_{3,21} = 4.36$, $P = 0.016$) which indicated that presenting male zebra finches with different stimuli evoked changing patterns of somatic investment. Body masses were lowest during the female conspecific treatment and highest during the 'no bird' stimulus treatment. The overall order of body masses in terms of stimulus treatment, from lowest to highest was; female zebra finch, male zebra finch, female bengalese finch and 'no bird' (figure 2).

The relative shapes of the daily mass trajectories of males were highly consistent between the stimulus treatments (treatment*time, $F_{81,567} = 0.64$, $P = 0.99$). The effect of treatment was manifested as a shift of mass trajectory in the y-axis as opposed to any change in shape of the trajectory. There was a highly significant effect of time of day on individuals' body mass (time, $F_{27,189} = 30.69$, $P < 0.0001$) indicating simply that body mass changed over the course of the day.

There was a significant effect of stimulus treatment on within-individual changes in hopping rates (treatment, $F_{3,21} = 3.13$, $P = 0.047$) with the highest hopping rates observed under the female zebra finch treatment (fig. 3). The hopping activity of male birds, independent of stimulus treatment, varied significantly over the course of the day (time, $F_{27,189} = 4.96$, $P<0.0001$) with a tendency for peaks to occur around dawn and dusk.

The temporal organisation of hopping behaviour varies significantly between treatments (treatment*time, $F_{81,567} = 1.305$, $P = 0.047$). Under the female zebra finch treatment there was a pronounced mid-morning peak whereas under the other treatments the level of hopping was relatively flat and uniform until just prior to dusk. At dusk the hop rates under all treatments increased steeply until total darkness fell and activity ceased.

The results of the specific contrasts (for mass and hop rate) which were tested are given in table 1. There were significant differences, for both body mass and hop rate, between the effect due to a female zebra finch versus any bird (treatment 1 included) and the effect due to a female zebra finch versus no bird.

Discussion

The opportunity to court had significant effects on the directed hopping activity trajectories, with the female zebra finch treatment differing markedly from the others. That trajectory showed the sharpest rise in activity from first light leading to a distinct peak 2 hours after dawn. The level of directed hopping during this treatment then remained highest overall through until 1 hour before last light when activity became the lowest overall. With the possible exception of the absence of a pronounced second peak at dusk, the shape of that hopping trajectory is otherwise very typical of male temporal patterns of display (chapters 3, 5, 6 & 7 of this thesis) and both equivalent and analogous to the patterning of song output in passerines (Kacelnik & Krebs, 1982; McNamara et al., 1987; Cuthill & MacDonald, 1990; Hutchinson et al., 1993; Pinxten & Eens, 1998). It therefore appears that the males responded to the presence of a female conspecific by performing typical diel patterns of courtship.

Hopping rates during the other three treatments were not significantly different from one another when compared pairwise, and were on average 37.2% lower than those seen in the female zebra finch treatment. Even in the absence of a stimulus bird a base-line level of hopping activity was expected due to aural stimuli from the other experimental cages (e.g. vocalizations associated with male-female interactions and actuation 'clicks' of microswitch perch units during display bouts elsewhere). There is evidence that male-male interactions evoke courtship song (McNamara et al., 1987) and in zebra finches that male intrasexual interactions lead to modulation of courtship behaviour (Zann, 1996; Waas & Wordsworth, 1999).

Given the differences in mean hopping rates and hopping trajectories between treatments, it is notable that the mass regulation trajectories did not also vary between treatments. Instead, the effect of treatment was evident as a complete shifting of the mass trajectory in the y-axis, with the highest masses during the 'no-bird' treatment and the lowest (between 0.16 and 0.31 grams less) during the female zebra finch treatment. Small passerines exhibit a predictable mass regulation curve whereby body mass typically increases most rapidly from dawn to mid-morning at which point the rate of increase slows down, plateaus or even declines, followed by another gain in mass

during the period from late afternoon until dusk (Owen, 1954). These curves have frequently been predicted by models (e.g. McNamara et al., 1987; Hutchinson et al., 1993) and shown by experimental studies (e.g. Lilliendahl et al., 1996; Dall & Witter, 1998; Pravosudov & Grubb, 1998; Cuthill et al., 2000). In all four treatments the body mass trajectories follow this typical curve with a very high degree of consistency.

This consistency in body mass trajectories coupled with very different display hopping trajectories could be explained in a number of ways. Males may have been compensating for their increased activity and guarding their diel mass trajectory, by strategically increasing their food intake at those times of day when energy expenditure was greatest. Time was highly unlikely to be a limiting factor here: a similar study on captive zebra finches indicated that 30% to 50% of time was spent inactive (Dall & Witter, 1998) while in captive starlings (Sturnus vulgaris) more than 90% of time was spent resting (Bautista et al., 1998). As food was in *ad libitum* supply in the present experiment it could not have been a limiting factor. Although mean body mass could have been maintained by balancing increased expenditure with increased food intake there are reasons why this strategy may not have been adopted. Owing to the design of the experimental cages, foraging necessitated letting the female out of sight, which would have punctuated the courtship routine and left her unguarded (Mace, 1989; Møller, 1991). An added disincentive is that any strategic variation in food intake would, *de facto*, have involved spending most time away from the female at those specific points in time when directed hopping activity and the context for courtship to take place would have been at their highest. Despite the laboratory setting, the birds may also have been taking time-dependent risks associated with foraging into account, for example exposure to predators, pathogens or parasites (Deerenberg et al., 1998). Additional foraging effort may also have had detrimental effects in terms of long term fitness (Lemon, 1993) due to hypertrophy of the gut, tissue damage or metabolic dysfunction (Sjödin et al., 1990).

Assuming that neither food nor the time to ingest it were in limited supply, there may have been a digestive bottleneck which prevented individuals from processing food any more rapidly (Sibly & Calow, 1986). Such an effect has been shown in juvenile house sparrows (Passer domesticus), a passerine of similar size to the zebra finch, where following restricted feeding individuals failed to process *ad libitum* food at an increased

rate (Lepczyk et al., 1998; Lepczyk & Karasov, 2000). If the reduction in body mass experienced by the harder working birds under the female zebra finch treatment was accompanied by a decrease in gut mass then such a digestive bottleneck would become ever more constricted and therefore be self-reinforcing. However, other work by the authors (chapter 7) showed that male zebra finches under similar conditions had spare digestive capacity which enabled them to more than double their rate of gain of body mass following a period of food restriction, despite a compromised body mass.

Studies on zebra finches (Deerenberg et al., 1998) and starlings (Bautista et al., 1998) have demonstrated that individuals subjected to harder work regimes actually had lower daily energy intakes and lower energy consumption, together with lower body masses but similar mass trajectories. The reduced energy consumption in these instances is largely attributed to reductions in nocturnal energy consumption, due to lower mass-specific costs or perhaps hypothermia, which may also have carried over into daylight hours. Other ways in which energy was conserved was by increasing the nutrient usage of food (as indicated by the calorific density of faeces) and by performing tasks (or behaviours) in a more efficient manner. More recently Swaddle & Biewener (2000) have demonstrated that starlings that were trained to fly more decreased their mass and particularly their muscle mass.

Therefore an alternative explanation for the consistent mass trajectories seen here despite the different courtship patterns is that the increased energy expenditure under the female zebra finch treatment was paid for by a decrease in the mass-specific costs of locomotion, metabolism and/or thermostasis, brought about by an overall reduction in body mass (Deerenberg et al., 1996; Bautista et al., 1998; Swaddle & Biewener, 2000). In addition it is possible that thermostatic costs were being met to a greater extent by heat generated as a consequence of the additional activity as opposed to metabolic activity expressly for that purpose. Every metabolic process has a ceiling above which it cannot ordinarily rise and these ceilings depend upon whether activity is performed in short bursts or sustained over a period of time (Suarez, 1996). If the males' average metabolic rates were running near to their aerobic limit, one means of supporting above-average activity without resorting to anaerobiosis would be to reduce metabolic overheads by reducing body mass. This hypothesis takes account of the fact that body masses were consistently lower throughout the day under the female zebra finch

treatment, a fact that is somewhat at odds with any explanation exclusively involving compensatory foraging.

Whichever the mechanism or mechanisms by which body mass trajectory is maintained despite the variation in courtship hopping activity, there is clear evidence of a short-term dynamic trade off between reproductive effort and body mass regulation as predicted by theory but shown here empirically for the first time.

References

Bautista, L. M., Tinbergen, J., Wiersma, P. & Kacelnik, A. 1998. Optimal foraging and beyond: how starling cope with changes in food availability. American Naturalist, 152, 543-561.

Bednekoff, P. A. & Houston, A. I. 1994. Avian daily foraging patterns: effects of digestive constraints and variability. Evolutionary Ecology, 8, 36-52.

Bednekoff, P. A. & Krebs, J. R. 1995. Great Tit fat reserves: effects of changing and unpredictable feeding day length. Functional Ecology, 9, 457-462.

Bednekoff, P. A. 1996. Translating mass dependent flight performance into predation risk: an extension of Metcalfe & Ure. Proceedings of the Royal Society of London Series B, 263, 887-889.

Blem, C. R. 1976. Patterns of Lipid Storage and Utilization in Birds. American Zoologist, 16, 671-684.

Cresswell, W. 1998. Diurnal and seasonal mass variation in blackbirds Turdus merula: consequences for mass dependent predation risk. Journal of Animal Ecology, 67, 78-90.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Cuthill, I. C., Maddocks, S. A., Weall, C. V. & Jones, E. K. M. 2000. Body mass regulation in response to changes in feeding predictability and overnight energy expenditure. Behavioral Ecology, 11, 189-195.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Deerenberg, C., Overkamp, G. J. F., Visser, G. H. & Dann, S. 1998. Compensation in resting metabolism for experimentally increased activity. Comparative Physiology B, 168, 507-512.

Dunn, A. M. & Zann, R. A. 1996a. Undirected song encourages the breeding female zebra finch to remain in the nest. Ethology, 102, 540-548.

Dunn, A. M. & Zann, R. A. 1996b. Undirected Song in Wild Zebra Finch Flocks: Contexts and Effects of Mate Removal. Ethology, 102, 529-539.

Ekman, J. B. & Hake, M. K. 1990. Monitoring starvation risk: adjustments of body reserves in greenfinches (Carduelis chloris L.) during periods of unpredictable foraging success. Behavioral Ecology, 1, 62-67.

Even, P. C. & Nicolaïdis, S. 1993. Adaptive changes in energy expenditure during mild and severe feed restriction in the rat. Journal of Nutrition, 70, 421-431.

Gosler, A. G., Greenwood, J. J. D. & Perrins, C. 1995. Predation risk and the cost of being fat. Nature, 377, 621-623.

Guntrip, J., Sibly, R. M. & Smith, R. H. 1997. Controlling resource acquisition to reveal a life history trade-off: egg mass and clutch size in an iteroparous seed predator, Prostephanus truncatus. Ecological Entomology, 22, 264-270.

Hainsworth, F. R. 1978. Feeding: Models of Costs and Benefits in Energy Regulation. American Zoologist, 18, 701-714.

Hiebert, S. M. 1991. Seasonal differences in the response of Rufous Hummingbirds to food restriction: body mass and the use of torpor. The Condor, 93, 526-537.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

King, J. R. & Farner, D. S. 1966. The adaptive role of winter fattening in the white-crowned sparrow with comments on its regulation. American Naturalist, 100, 403-418.

Kokko, H. 1998. Good genes, old age and life-history trade-offs. Evolutionary Ecology, 12, 739-750.

Lehikoinen, E. 1986. Winter ecology of passerines: significance of weight and size. PhD thesis. University of Turku.

Lemon, W. C. 1993. The energetics of Lifetime Reproductive Success in the Zebra Finch Taeniopygia guttata. Physiological Zoology, 66, 946-963.

Lepczyk, C. A. & Karasov, W. H. 2000. Effect of ephemeral food restriction on growth of house sparrows. The Auk, 117, 164-174.

Lepczyk, C. A., Caviedes-Vidal, E. & Karasov, W. H. 1998. Digestive responses during food restriction and realimentation in nestling house sparrows (Passer domesticus). Physiological Zoology. 71, 561-573.

Lilliendahl, K., Carlson, A., Welander, J. & Ekman, J. B. 1996. Behavioural control of daily fattening in great tits (Parus major). Canadian Journal of Zoology, 74, 1612-1616.

Mace, R. 1989. Great tits choose between food and proximity to a mate: the effect of time of day. Behavioural Ecology and Sociobiology, 24, 285-290.

Mace, R.1987. The dawn chorus in the great tit Parus major is directly related to female fertility. Nature, 330, 745-746.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Metcalfe, N. B. & Ure, S. E. 1995. Diurnal variation in flight performance and hence potential predation risk in small birds. Proceedings of the Royal Society of London Series B, 261, 395-400.

Metcalfe, N. B., Fraser, N. H. C. & Burns, M. D. 1998. State-dependent shifts between nocturnal and diurnal activity in salmon. Proceedings of the Royal Society of London Series B, 265, 1503-1507.

Møller, A. P. 1991a. Why mated songbirds sing so much: mate guarding and male announcement of mate fertility status. American Naturalist, 138, 994-1014.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Owen, D. F. 1954. The winter weights of titmice. Ibis, 96, 299-309.

Pinxten, R. & Eens, M. 1998. Male starlings sing most in the late mornings, following egg-laying: a strategy to protect their paternity? Behaviour, 135, 1197-1211.

Pravosudov, V. V. & Grubb, T. C. 1998. Management of fat reserves in tufted titmice (Parus bicolor): evidence against a tradeoff with food hoards. Behavioural Ecology and Sociobiology, 42, 57-62.

Rogers, C. M. & Smith, J. N. M. 1993. Life-history theory in the nonbreeding period-tradeoffs in avian fat reserves. Ecology, 74, 419-426.

Sibly, R. M. & Calow, P. 1986. Physiological Ecology. Blackwell Scientific, Oxford.

Sjödin, B., Westing, Y. H. & Apple, F. S. 1990. Biochemical mechanisms for oxygen free radical formation during exercise. Sports Medicine, 10, 236-251.

Stearns, S. C. 1989. Trade-offs in life-history evolution. Functional Ecology, 3, 259-268.

Suarez, R. K. 1996. Upper limits to mass-specific metabolic rates. Annual Review of Physiology. 58, 583-605.

Swaddle, J. P. & Biewener, A. A. 2000. Physiology- Exercise and reduced muscle mass in starlings. Nature, 406, 585-586.

Swaddle, J. P. & Cuthill, I. C. 1994. Preference for symmetric males by female zebra finches. Nature, 367, 165-166.

Veasey, J. S., Metcalfe, N. B. & Houston, D. C. 1998. A reassessment of the effect of body mass upon flight speed and predation risk in birds. Animal Behaviour, 56, 883-889.

Waas, J. R. & Wordsworth, A. F. 1999. Female zebra finches prefer symmetrically banded males, but only during interactive mate choice test. Animal Behaviour, 57, 1113-1119.

Witter, M. S. & Cuthill, I. C. 1993. The ecological costs of avian fat storage. Philosophical Transactions of the Royal Society of London B, 340, 73-92.

Witter, M. S., Cuthill, I. C. & Bonser, R. H. C. 1994. Experimental investigations of mass-dependent predation risk in the European starling, Sturnus vulgaris. Animal Behaviour, 48, 201-222.

Workman, L. & Andrew, R. J. 1986. Asymmetries of eye use in birds. Animal Behaviour, 34, 1582-1584.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Zar, J. H. 1996. Biostatistical Analyses. 3rd Edition. New Jersey: Prentice Hall International Inc.

Table 1. Results of specific contrasts. There were significant differences in both mean male body mass and mean male hopping activity when the female zebra finch treatment was compared to the other three treatments collectively (any bird), and when the female zebra finch treatment was compared to the 'no bird' treatment.

Description	*Mass*	*Hop Rate*
♀ conspecific *v* any bird	$F_{1,7} = 16.990$ $p = 0.004$*	$F_{1,7} = 8.081$ $p = 0.025$*
♀ conspecific *v* no bird	$F_{1,7} = 18.075$ $p = 0.004$*	$F_{1,7} = 7.573$ $p = 0.028$*

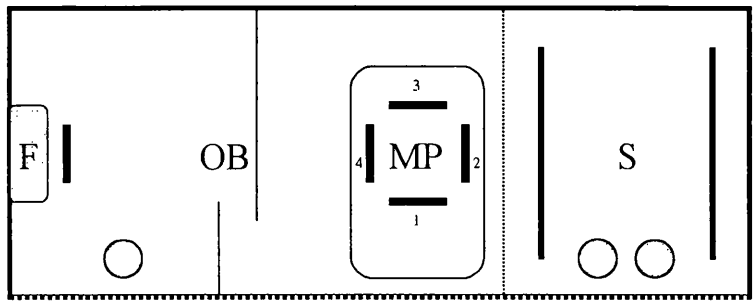


Figure 1. The experimental cage viewed from above. F = feed hopper, OB = opaque baffles, MP = microswitch perches mounted on electronic balance, S = stimulus cage. Heavy bars represent perches, dotted lines represent wire cage front and Perspex panel separating courtship and stimulus compartments. Shaded circles are food and water dishes.

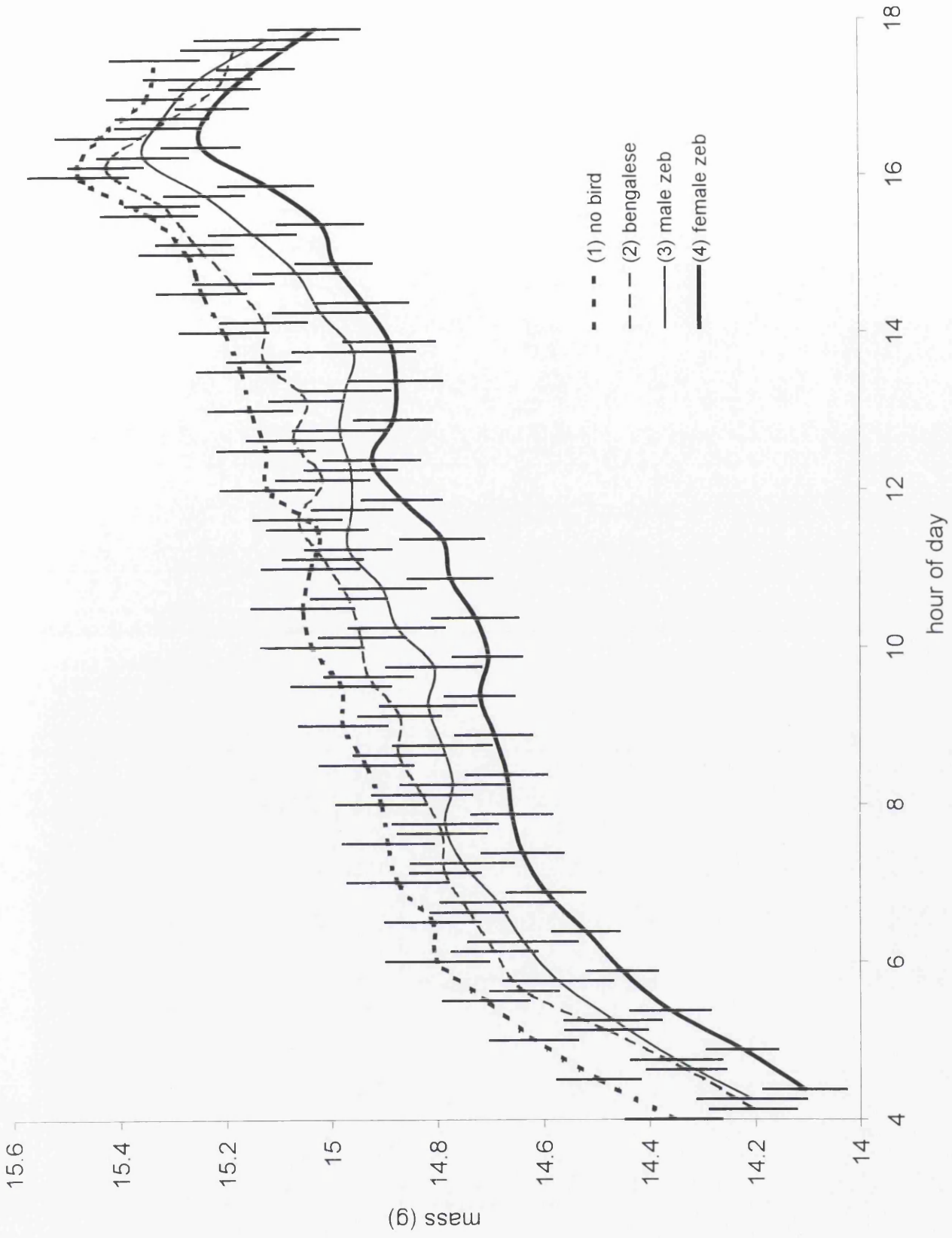


Figure 2. Mean body mass trajectories (±S.E.) of male zebra finches for each of the four treatments. X-axis values are staggered among treatments by 7.5 minutes for clarity (to prevent error bars from overlapping). Treatment 1, 'no bird'; Treatment 2, female bengalese finch; Treatment 3, male zebra finch; Treatment 4, female zebra finch.

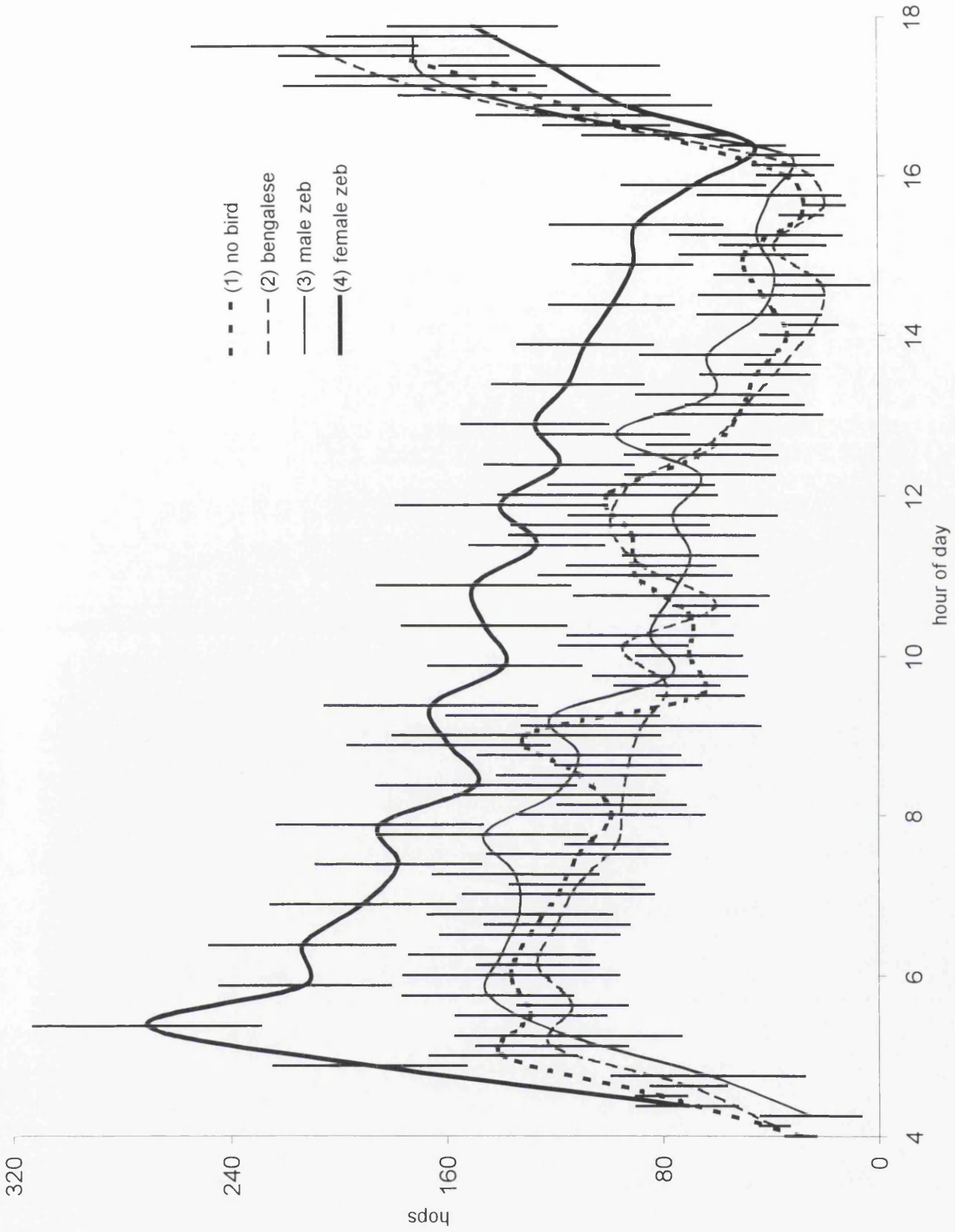


Figure 3. Mean hop rate trajectories (±S.E.) of male zebra finches for each of the four treatments. Data presented as in Fig. 2.

Chapter 5. The effects of time of day and context of female presentation on body mass and display in male zebra finches

Abstract

The often-observed diel variation in male display activity in birds could be due to temporal variation in either the state of the male, or in the receptivity and/or attractiveness of the female. In this experiment we investigated these effects by presenting male zebra finches (Taeniopygia guttata) with a female for discrete three hour periods at dawn and midday. We removed the association of the time-state of the two birds by desynchronising the female relative to the males' photoperiod, such that they were in either a dawn or a midday state at each of the times of day when presented to the male. Our analysis investigated the relative importance of the time of presentation versus the time-state of the female to male body mass regulation and courtship activity patterns. We found that both the time of presentation of a female and her time-state affected the display activity and mass regulation of male zebra finches, but in quite different ways. The time of day of presentation had no significant effect on overall male display rates, due we believe to the discrete presentation period and the opportunistic breeding nature of the species. However, we found that heightened display activity at dawn had the effect of significantly reducing overall body mass compared to if that same level of activity occurred at midday. We attribute this to the enormous disparity in male state between dawn and midday and suggest that lowered body mass was a means of reducing mass-dependent costs in order for a male to finance energetically expensive activity at the lowest point in his diurnal mass cycle when he had the lowest food/fat reserves. Female time-state (independent of male time-state) has for the first time been shown to influence display activity of males. This effect was context dependent insofar as dawn-state females were only more attractive and/or responsive to males during the males' dawn. Female time-state did not significantly affect the overall body mass of males although it did have significant effects on their body mass trajectory and we propose that these changes were mediated by the associated change in display pattern.

Introduction

The majority of passerines spend a higher proportion of their time singing at dawn compared with other times of the day (Kacelnik & Krebs, 1982). This phenomenon, known as the dawn chorus, has attracted a great deal of research into its cause and functions (e.g. Mace, 1987a, 1987b, 1989; McNamara et al., 1987; Cuthill & MacDonald, 1990; Staicer et al., 1996; Thomas, 1999a). The reason why birds choose to spend more time singing at dawn remains a subject of debate. Singing is a costly behaviour, but the net costs may vary with time of day. For example, due to environmental conditions early in the day, sound may carry further which would generally be of benefit to the singer (Henwood & Fabrick, 1979). Kacelnik (1979) suggested that foraging may be less profitable early in the day owing to reduced light intensity and therefore it was more efficient to sing, and postpone feeding until later in the day. Causal and functional explanations may be intertwined, for example Mace (1987a) found a direct relationship between female fertility and male song output at dawn, suggesting a role of the dawn chorus in mate guarding in response to heightened female fertility. There are occasions when the function of dawn choruses cannot be linked to mate guarding, for example in unpaired males or those of polygynous species (see Cuthill & MacDonald, 1990); this favours a more general explanation stemming from optimal policies and diurnal variation in costs.

Theoretical models, constructed using stochastic dynamic programming, have sought to explain the dawn chorus by weighing up the costs of singing versus foraging relative to male state, starvation risk and environmental stochasticity (Houston & McNamara, 1987; McNamara et al., 1987; Hutchinson et al., 1993). These models have demonstrated that the dawn chorus could be generated by variation in overnight energy expenditure irrespective of any other daily patterns in the environment, for example female fertility. Empirical tests of these models have shown that the timing and duration of the dawn chorus is linked to a males nutritional state and ambient temperature, in addition to his mate's fertility (Cuthill & MacDonald, 1990, Thomas, 1999a, 1999b). Although the theory has been developed with singing it mind, it can equally be applied to any mate attracting behaviour as long as it both consumes energy and is incompatible with foraging, criteria fulfilled by the display hopping of male zebra finches associated with stage 1 & 2 courtship routines of that species (see Zann, 1996).

It has already been shown that male zebra finches are able to assess aspects of female state and/or attractiveness and vary their behaviour accordingly. Monaghan et al. (1996) provided clear evidence that males were able to recognise more fecund females and selectively pair with them. In the present experiment we consider the effects of the time of presentation of females and female temporal state on the diel patterns of display activity and body mass regulation in male zebra finches.

Methods

We conducted this experiment using a group of eight male and a separate group of eight female zebra finches. The two groups were obtained from different suppliers and had no prior experience of each other. Males birds wore a single numbered orange leg band and females wore none. All birds were sexually mature and the females used were known to be between 2 and 4 years of age. Sexes were housed in visual but not aural isolation prior to the experiment and during the intervening periods between treatments. The prevailing photoperiod was 12:12 L:D (without simulated dawn or dusk, see chapter 3) with daylight completely excluded from the laboratory. Room temperature was kept relatively constant at mean ± S.D. = 21.0 ± 3.1°C by means of thermostatically controlled heating and ventilation. Birds were provided with *ad libitum* water and food in the form of a proprietary seed mix. In addition they were provided with shell grit and cuttlebone.

Eight experimental cages (1.5 x 0.3 x 0.4 m high, Figure 1) were employed, each consisting of three equally sized sections: food, courtship and stimulus. The experimental male had access to the food and courtship sections and the stimulus section was occupied periodically by the female zebra finch stimulus. The food and courtship sections were separated by opaque baffles which permitted the male to move freely from one section to the other but prevented him from simultaneously feeding and being in visual contact with the stimulus section. The courtship and stimulus sections were separated by a transparent Perspex panel (0.15 cm thick). The food section contained a food hopper and water dish together with a microswitch feeding perch that recorded the time and duration of feeding bouts (provided that the male chose to feed

from this perch). The courtship section contained a set of four microswitch perches that recorded the temporal and spatial patterns of hopping activity and were mounted on an electronic balance. The four perches, made from 5 cm lengths of wooden dowel, were arranged in a square with opposite pairs set 12 cm apart and each perch 14 cm above the cage floor.

The stimulus section was equipped with two perches set at the same height as the microswitch perches. The wall of the right-hand subsection against which female stimulus birds were viewed was white. The stimulus section was bedded with composted wood bark. The food and courtship sections were not bedded to prevent excessive debris being deposited by the males on either the perch mechanisms or the balances.

The experimental protocol involved using stimulus females whose time-states had been manipulated according to four treatments (see table 1). Females were presented to the males for discrete 3 hour periods corresponding to prevailing dawn or midday at which time the female's own internal clock was set to either dawn or midday. During treatments, each lasting five days, females were presented to all eight males in the treatment order 1, 2, 3 then 4 (table 1). The order of presentation was not randomised due to the practical considerations of desynchronising the females (see below), however, to reduce any carry-over effects males were allowed six days rest between treatments. Males were moved from their housing cages into the experimental cages one day prior to the first day of each treatment and all birds were returned to their housing cages on the day following the last day of each treatment. In the case of treatments presented at dawn, females were moved from their housing cage into the experimental cages immediately after the laboratory lights came on.

Except when being presented as stimuli, females were kept in a specially constructed light-sealed cage which had its own built-in lighting and ventilation. This enabled the females' time frame to be manipulated independently of the prevailing photoperiod. To avoid light contamination, all maintenance of this cage, and moving of birds, took place when internal and external environments were both experiencing 'daylight'. Females remained on a 12:12 photoperiod irrespective of treatment. The transition of females

from one time frame to another was introduced gradually over two days and females were then given a further four days to adjust before the next treatment began.

We investigated the effect of treatment on within-individual variation in body mass and hop rate using repeated measures ANOVA designs in SPSS 8.0.0. In addition we investigated the effect of treatment on the diurnal mass trajectory and organisation of hopping behaviour by considering the interaction between treatment and time of day (treatment*time). The data for each individual were averaged across the five days of each treatment. Body mass and display hopping data were condensed into mean mass per half hour interval and total number of hops per half hour (i.e. hop rate) over the course of each day.

More specifically, we looked at contrasts between pairs of treatments to examine the effects of female temporal state (treatments 1&3 versus 2&4) as opposed to the time of presentation of female (1&2 versus 3&4) on male body mass and hopping activity. This allowed us to achieve the core objective of this experiment which was to disentangle the effects of the time of presentation and time-state of the female on the diel mass regulation and display strategies of male birds.

All repeated measures ANOVA statistics given are univariate and where assumptions about sphericity were violated the Huynh-Feldt Epsilon corrected statistic is quoted. Two-tailed tests of probability are used throughout.

Results

Body mass regulation

The statistics showed a significant effect of treatment on male body mass (Figure 2; $F_{3,21} = 10.971$, $P < 0.0001$). Figure 2 indicates that the lowest overall male body masses were under those treatments where the female was presented at dawn while the higher male body masses were under those where she was presented at midday. There was also a highly significant effect of the treatment*time interaction term on body mass ($F_{69,483} = 3.417$, $P < 0.0001$) which indicated that the shapes of the daily body mass curves (trajectories) varied between treatments (see Figure 2).

The contrast to investigate the effect of time of presentation period on male mass (treatments 1&2 versus 3&4) revealed a significant effect (Figure 3; $F_{1,7} = 19.134$, $P = 0.003$), again with lower masses during the 'female presented at dawn' treatments and higher masses during the 'female presented at midday' treatments. This result indicated that the time of presentation (i.e. an effect linked to the time-state of the male) had a significant effect on his body mass regulation, independently of how the female perceived the time of day. The effect of presentation period on mass trajectory was not quite significant at $P = 0.079$ ($F_{1.70,11.90} = 3.29$).

A second contrast tested the effect of female temporal state on male mass regulation (treatments 1&3 versus 2&4). There was no significant effect on male mass *per se* (Figure 4; $F_{1,7} = 2.647$, $P = 0.15$) although the tendency was for males to be lighter during the dawn-state female treatments. There was, however, a significant effect on body mass trajectory (female time state*time: $F_{3.95,27.64} = 3.958$, $P = 0.012$) manifested as a plateau around midday followed by a sharp rise towards dusk under the treatments where the female considered it to be around midday when she met the male, compared to a more typical (near linear) curve under the dawn-state female treatments.

Display hopping activity

There was no effect of treatment on overall hopping activity (Figure 5; $F_{3,21} = 1.372$, $P = 0.28$), with each of the four treatments showing a similar mean hopping rate. There was however a strong effect of treatment*time on hopping activity ($F_{69,483} = 3.414$,

P<0.0001) indicating that the daily organisation of display activity differed between treatments, with hopping activity peaking at different times of day corresponding to the discrete period when the female stimulus was present.

The contrast to investigate the effect of presentation period of females to males indicated no effect on male hopping activity (Figure 6; $F_{1,7} = 0.402$, $P = 0.55$) but there was a significant effect on the temporal patterning of hopping activity (presentation period*time: $F_{2.25,15.74} = 4.263$, $P = 0.03$) caused by display peaks which were very similar in amplitude but occurred during their respective presentation period. This indicated that males, when presented with a female for a discrete three hour period, behaved opportunistically and displayed equally to her irrespective of whether it was dawn or midday.

The second contrast showed a significant effect of female state on hopping activity (Figure 7; $F_{1,7} = 5.724$, $P = 0.048$) with greater display effort by the male occurring at dawn towards a female who considered it to be dawn, as opposed to a female who considered it to be midday. This indicated that females who considered it to be dawn were more attractive and/or responsive to the males, but only in the context of the male's dawn. The effect on the temporal pattern of hopping activity was not significant (female state*time: $F_{2.64,18.47} = 1.256$, $P = 0.32$).

Discussion

The results showed that the time of presentation coupled with the temporal state of the presented female had significant effects on male body mass and mass regulation strategy, and while overall hopping activity did not differ there were significant effects of treatment on hopping activity patterns over time of day. The core issue however was to disentangle the effect of the time-state of the female zebra finch from that of the time of day at which she was presented. To explore this question we looked at pairwise contrasts between treatments.

We found no effect of presentation time on overall hopping activity, which in both treatments (dawn and midday presentation time) was consistent in peaking at around

225 hops per half hour (figure 6). In passerines in general, dawn (and to a lesser extent dusk) is the period of the day when courtship activities are at their peak and towards midday courtship activity begins to decline markedly. Such routines have been consistently demonstrated in zebra finches (Dall & Witter, 1998; chapters 3, 4, 6 & 7 of this thesis) but in these cases the opportunity for interaction between the male and female was not restricted to discrete times as in the present study. Given that in the present study, daily courtship was limited to a three hour period, it is unsurprising that a strongly opportunistic breeder like the zebra finch (Zann, 1996) opted to maximise that opportunity whether it occurred at dawn or midday. Despite these similar peaks of activity at dawn and at midday there was a significant disparity in mean body masses which were between 2.7% and 5.9% lower during the dawn presentation treatments (figure 3). The reason behind this disparity may be related to the states of the males at dawn compared to midday. The costs of being fatter are well documented (Hainsworth, 1978; Witter & Cuthill, 1993; Gosler et al., 1995; Houston et al., 1997) and a key assumption of models of avian mass regulation is that individuals will seek to minimise their risk of overnight starvation by attaining an optimal body mass the previous dusk (Houston & McNamara, 1987; McNamara et al., 1987). In a stochastic environment individuals will budget for a worst-case scenario of overnight conditions. On those occasions therefore where the reality is better than the forecast, individuals will find themselves at dawn with energy reserves greater than zero. In this study, environmental conditions were stable and we would have expected target levels of dusk mass (relative to the mass of the individual) to remain approximately constant. Moreover, given the absence of any discernible stochasticity, theory would predict that the birds would arrive at dawn with energy reserves tending towards zero. We know from our measurements in general that body masses were always lowest at dawn which in turn meant that fat/food reserves (Gosler et al., 1995) and also mass-dependent costs were at their lowest. Following a 12 hour fast (during darkness), the gut contents would also have fallen to their lowest level and indeed it is highly likely that the gut would have been completely empty (Cade et al., 1965). In contrast, males at midday would have had intermediate body masses (which equated to intermediate fat/food reserves) and therefore an intermediate degree of mass-dependent costs. The gut content at midday would probably not have been full but certainly would have been substantially fuller than at dawn.

Despite the very different states of the males at dawn versus midday we measured near identical levels of display activity. Clearly, although the absolute costs of the display activity at dawn and at midday would have been very similar, the costs relative to male state must have been different. The significantly decreased body masses which accompanied the 'presented at dawn' treatment introduced an effect of treatment on male state whereby mass-dependent costs (Hainsworth, 1978; Witter & Cuthill, 1993; Gosler et al., 1995; Houston et al., 1997) would have been significantly lower during that treatment. Although there was no evidence in the present study that the overall workload varied between treatments, there was a disparity in male energetic state despite which a similar level of work was performed. Other studies have found relationships between increased workload and decreased body mass. Deerenberg et al. (1998) found that zebra finches that were required to work harder for food rewards actually had lower overall daily energy budgets coupled with lower body masses. The authors speculated that apart from a reduction of mass-specific costs, other mechanisms, for example nocturnal and even diurnal hypothermia, were employed to finance heightened workloads. Bautista et al. (1998) found a similar result using European starlings (Sturnus vulgaris) that were subjected to hard and easy foraging modes. Under the hard treatment the daily amount of work was greater but overall daily energy expenditure was lower, as too was body mass. Finally, Swaddle & Biewener (2000) demonstrated that starlings that were trained to fly more, decreased their body mass, specifically their muscle mass, which appeared to be a means of lowering mass-dependent costs of locomotion.

The significant decrease in body mass that occurred during the 'presented at dawn' treatments, and the concomitant mass-dependent energy saving, may be the way in which a significantly lighter bird, with low reserves and an initially empty crop and gut, financed the same level of activity as it achieved when heavier and fatter, while simultaneously maintaining the same mass trajectory. The benefit to the individual was the performance of a vigorous display at dawn with the reproductive advantages that may have brought (Cuthill & MacDonald, 1990), while the cost it paid was a heightened risk of starvation owing to a much reduced body mass (Houston & McNamara, 1987; McNamara et al., 1987; Witter & Cuthill, 1993).

When we considered the data in terms of the effect of female state (i.e. presenting her to the male when she was either in a dawn or a midday time-state) we found a significant effect on male display rate (figure 7) and also on male body mass trajectory (figure 4). Although there was no significant effect on overall mass per se, there was a tendency for males to be lighter under the dawn-state female treatments. The significant variation in body mass trajectories between female state treatments appeared to be due to the higher initial gain in male mass under the midday-state treatment that accompanied the reduced level of hopping during the first hour of the day. This initial gain was followed by a plateau around midday which differed from the more typical mass curve seen under the dawn-state treatment. The decrease in body mass between treatments, although not significant, could be viewed as analogous to the situation described for the effect of the time of presentation. That is to say, the increased display activity directed towards dawn-state females at dawn, by males in their lightest and least-fed state, was being financed by a sacrificial decrease in body mass which made additional energy available by reducing mass-dependent costs (Bautista et al., 1998; Deerenberg et al., 1998; Swaddle & Biewener, 2000).

The effect of dawn-state females on hopping activity was to make the dawn display effort higher in mean amplitude but otherwise, across the remainder of the day, hopping activity was near to identical between treatments. Courtship in zebra finches is a strongly interactive process, generally initiated by the male and thereafter involving stereotypical parallel hopping and hop-pivot dancing, much of which is reciprocated by the female (Morris, 1954, Immelmann, 1959 & 1962 as cited by Zann, 1996; Zann, 1996). Therefore, the role of the female is far from passive and her perceived attractiveness and/or responsiveness will influence the duration and amplitude of male display. Dawn is the natural time of day for courtship events (Kacelnik & Krebs, 1982; Mace, 1987a, 1987b,1989; Cuthill & MacDonald, 1990; Thomas, 1999a, 1999b) and this study has confirmed that females were more attractive and/or responsive to males when the females were in a dawn-state as compared to a midday-state. The relationship between attractiveness/responsiveness and female state would appear to be context dependent as there were apparently no differences in display effort between the two groups corresponding to the presentation at midday. We therefore surmise that a dawn-state female is only more attractive/responsive when presented in the context of the prevailing dawn.

In conclusion we found that there were effects of both presentation time and time-state of the female on diel patterns of display hopping and body mass regulation in male zebra finches. We surmise that the effects of presentation time on mass regulation are strongly related to differences in male state between dawn and midday- specifically that the costs of display relative to male state are much higher at dawn and consequently body mass is traded off to finance those costs. This study has also demonstrated that female time-state influences her attractiveness and/or responsiveness to courting males- an effect never previously demonstrated. Females in the dawn-state evoked a more sustained display effort from the males, but only when presented in the context of the males' dawn.

References

Bautista, L. M., Tinbergen, J., Wiersma, P. & Kacelnik, A. 1998. Optimal foraging and beyond: how starling cope with changes in food availability. American Naturalist, 152, 543-561.

Cade, T. J. & Tobin, C. A. & Gold, A. 1965. Water economy and metabolism of two estrilidine finches. Physiological Zoology, 38, 9-33.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Deerenberg, C., Overkamp, G. J. F., Visser, G. H. & Dann, S. 1998. Compensation in resting metabolism for experimentally increased activity. Comparative Physiology B, 168, 507-512.

Gosler, A. G., Greenwood, J. J. D. & Perrins, C. 1995. Predation risk and the cost of being fat. Nature, 377, 621-623.

Hainsworth, F. R. 1978. Feeding: Models of Costs and Benefits in Energy Regulation. American Zoologist, 18, 701-714.

Henwood, K. & Fabrick, A. 1979. A quantitative analysis of the dawn chorus: temporal selection for communicatory optimisation. American Naturalist, 114, 260-274.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Houston, A. I., Welton, N. J. & McNamara, J. M. 1997. Acquisition and maintenance costs in the long-term regulation of avian fat reserves. Oikos, 78, 331-340.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Immelmann, K. 1959. Exerimentelle untersuchungen über die biologische bedeutung artsspezifischer Merkmale beim Zebrafinken (Taeniopygia guttata). Zoologische Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 86, 437-592.

Immelmann, K. 1962. Beiträge zu einer vergleichenden Biologie australische Prachtfinken (Spermestidae). Zoologische Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 90, 1-196.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

Kacelnik, A. 1979. Studies on foraging behaviour and time budgeting in the great tit (Parus major). Unpublished D Phil Thesis, University of Oxford.

Mace, R. 1987. The dawn chorus: behavioural organisation in the great tit (Parus major). Unpublished D Phil Thesis, University of Oxford.

Mace, R. 1987. The dawn chorus in the great tit Parus major is directly related to female fertility. Nature, 330, 745-746.

Mace, R. 1989. Great tits choose between food and proximity to a mate: the effect of time of day. Behavioural Ecology and Sociobiology, 24, 285-290.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Monaghan, P., Metcalfe, N. B. & Houston, D. C. 1996. Male finches selectively pair with fecund females. Proceedings of the Royal Society of London Series B, 263, 1183-1186.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Staicer, C. A., Spector D. A. & Horn, A. G. 1996. The dawn chorus and other diel patterns in acoustic signalling. In: Ecology and Evolution of Acoustic Communication in Birds (Editors D.E. Kroodsma and E. H. Miller), pps. 426-453. Ithaca, New York: Cornell University Press.

Swaddle, J. P. & Biewener, A. A. 2000. Physiology- Exercise and reduced muscle mass in starlings. Nature, 406, 585-586.

Thomas, R. J. 1999a. Two tests of a stochastic dynamic programming model of daily singing routines in birds. Animal Behaviour, 57, 277-284.

Thomas, R. J. 1999b. The effect of variability in the food supply on the daily singing routines of European robins: a test of a stochastic dynamic programming model. Animal Behaviour, 57, 365-369.

Witter, M. S. & Cuthill, I. C. 1993. The ecological costs of avian fat storage. Philosophical Transactions of the Royal Society of London B, 340, 73-92.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Table 1. Description of the four treatments. In each case the male experienced dawn at 1000 hours, so that in treatment 2 he saw a female for the first three hours of his day, whereas from her perspective it was already midday. Conversely, in treatment 3, the male first encountered the female five hours after his dawn, but when she was just experiencing dawn.

Time of Presentation to Male		
Dawn (1000 to 1300 hours)	**Treatment 1 Dawn Female** Female experiences dawn at 1000 hours (not desynchronised)	**Treatment 2 Midday Female** Female experiences dawn at 0500 hours (desynchronised)
Midday (1500 to 1800 hours)	**Treatment 3 Dawn Female** Female experiences dawn at 1500 hours (desynchronised)	**Treatment 4 Midday Female** Female experiences dawn at 1000 hours (not desynchronised)

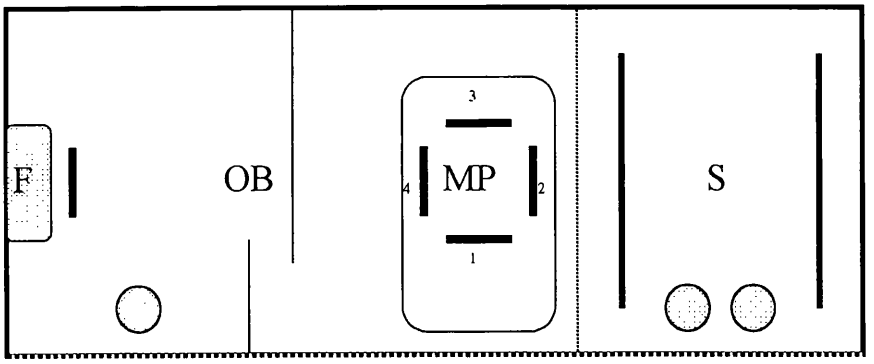


Figure 1. The experimental cage viewed from above. F = feed hopper, OB = opaque baffles, MP = microswitch perches mounted on electronic balance, S = stimulus cage. Heavy bars represent perches, dotted lines represent wire cage front and Perspex panel separating courtship and stimulus compartments. Shaded circles are food and water dishes.

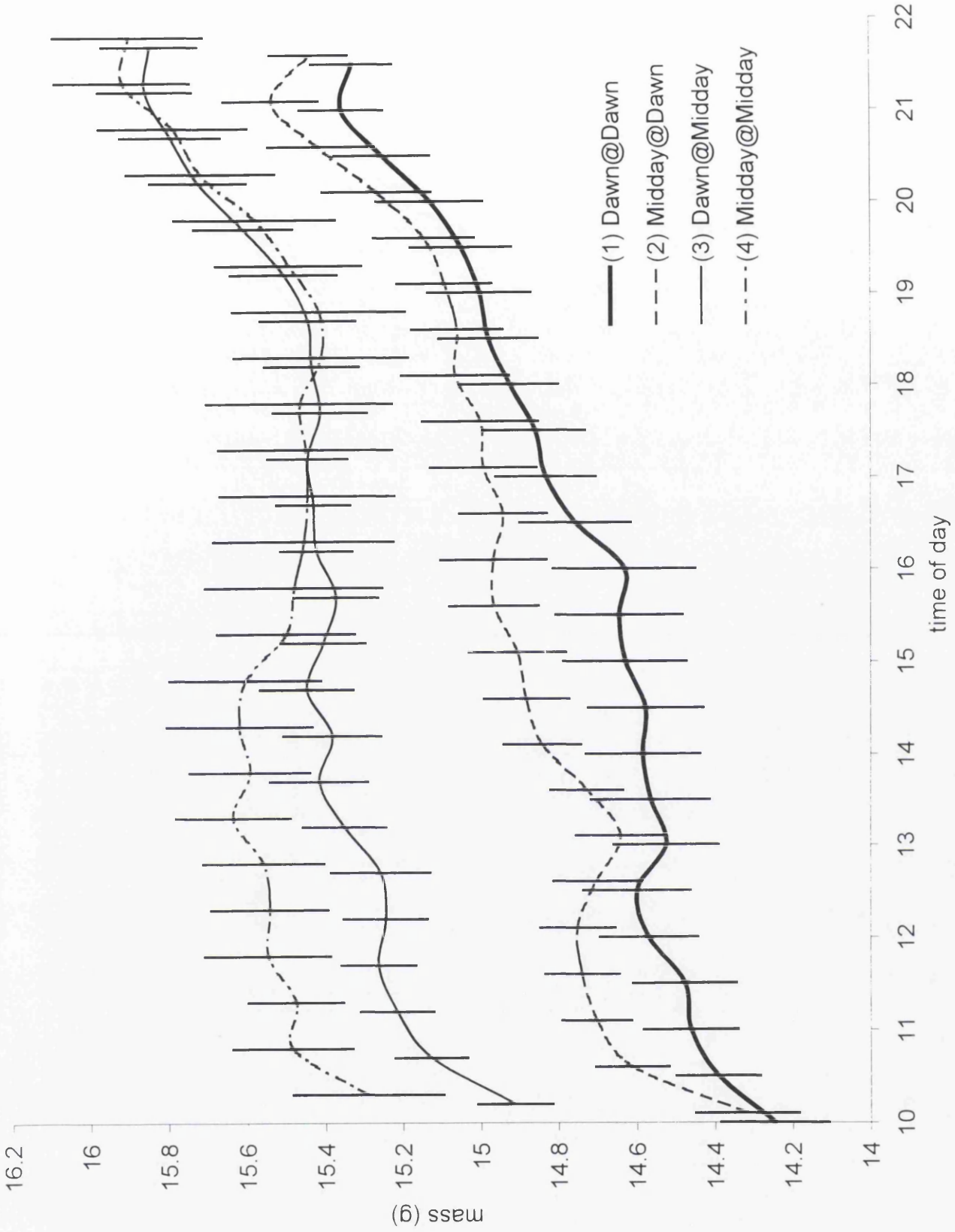


Figure 2. Mean body mass trajectories (± S.E.) of male zebra finches under four treatments, differing in both the time of day at which a female was presented and the time of presentation as perceived by the female (see text and table 1 for further clarification). X-axis values are offset between treatments by 6 minutes for clarity (to prevent error bars from overlapping). Treatment 1, dawn female zebra finch presented to the male at his dawn; Treatment 2, midday female presented at dawn; Treatment 3, dawn female presented at midday; Treatment 4, midday female presented at midday. Presentation periods were 1000 to 1300 hours (dawn) or 1500 to 1800 (midday).

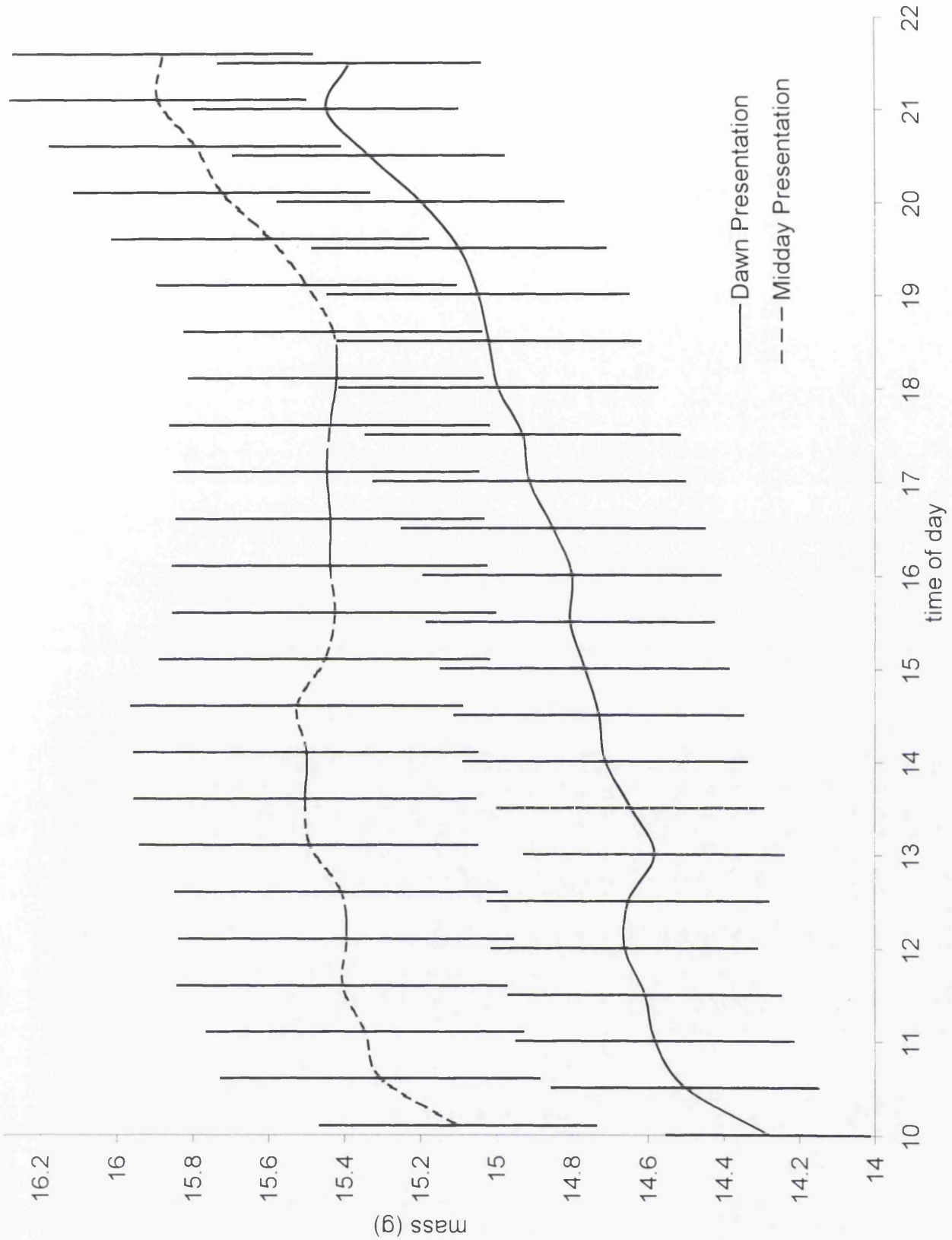


Figure 3. Mean body mass trajectories (± S.E.) of male zebra finches showing effect due to presentation time; Treatments 1&2 (female presented at dawn) versus 3&4 (female presented at midday). Otherwise data presented as in Figure 2.

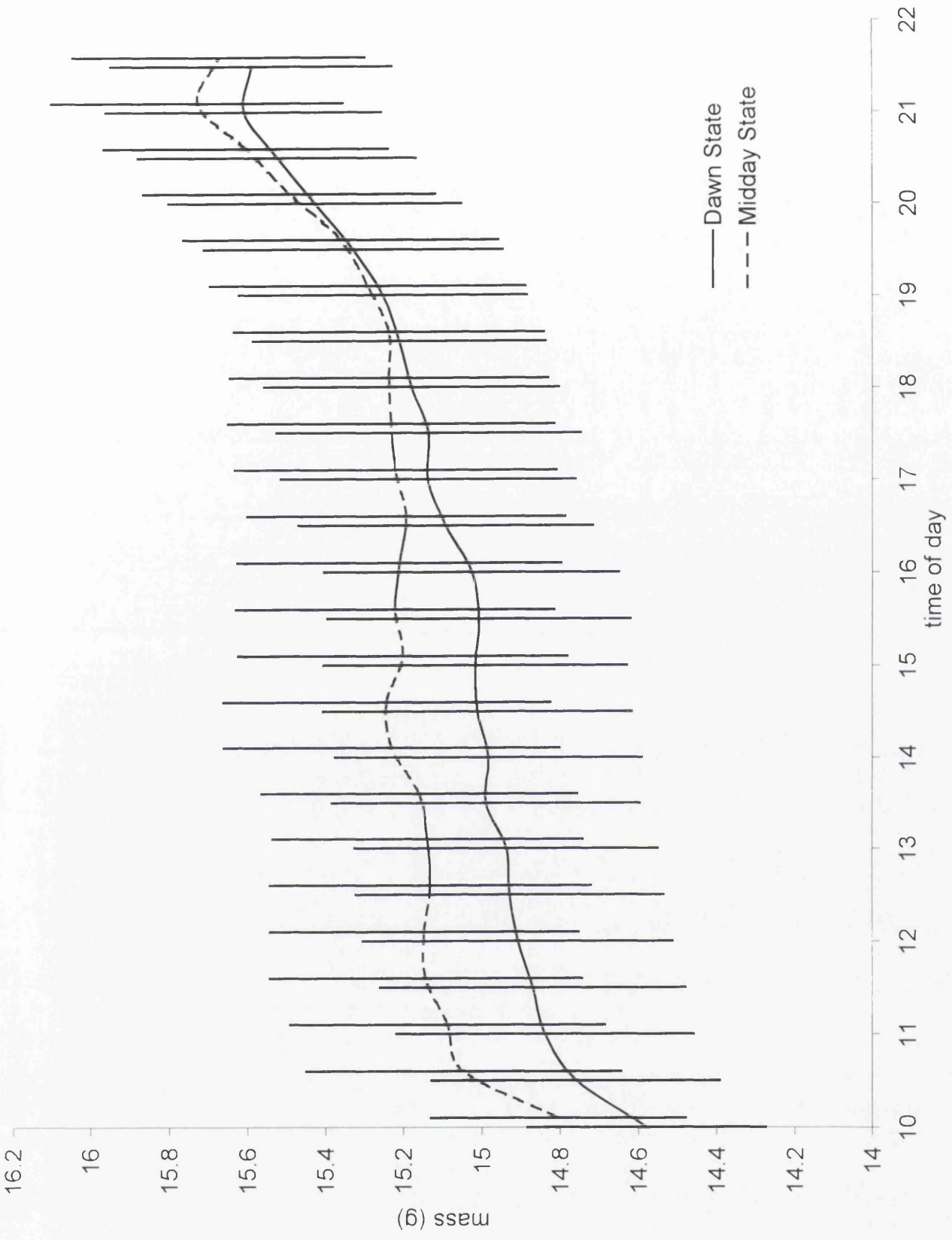


Figure 4. Mean body mass trajectories (± S.E.) of male zebra finches showing effect due to female state; Treatments 1&3 (female in dawn-state at time of presentation) versus 2&4 (female in midday-state). Otherwise data presented as in Figure 2.

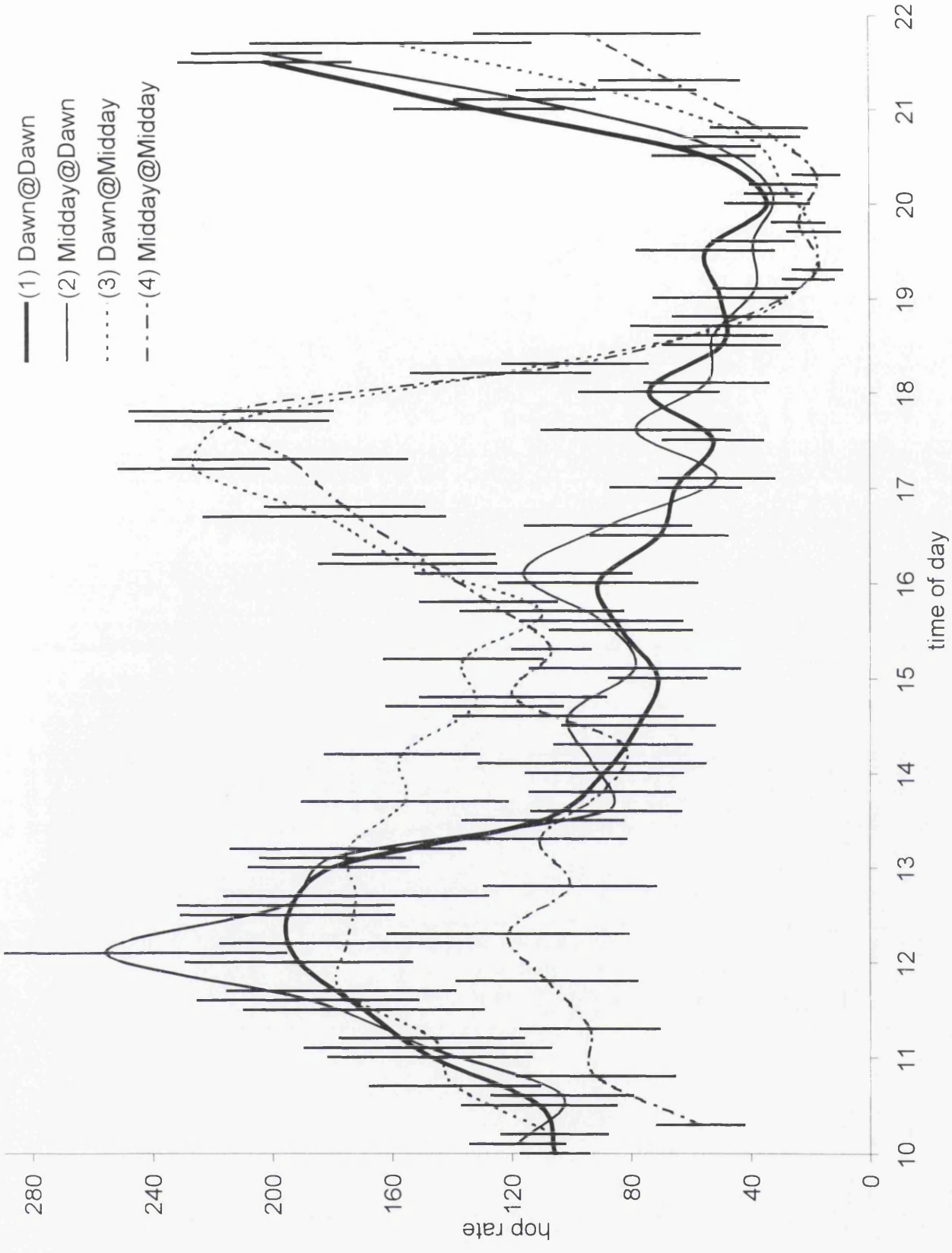


Figure 5. Mean hop rate trajectories (± S.E.) of male zebra finches under the four treatments. Data presented as in Figure 2.

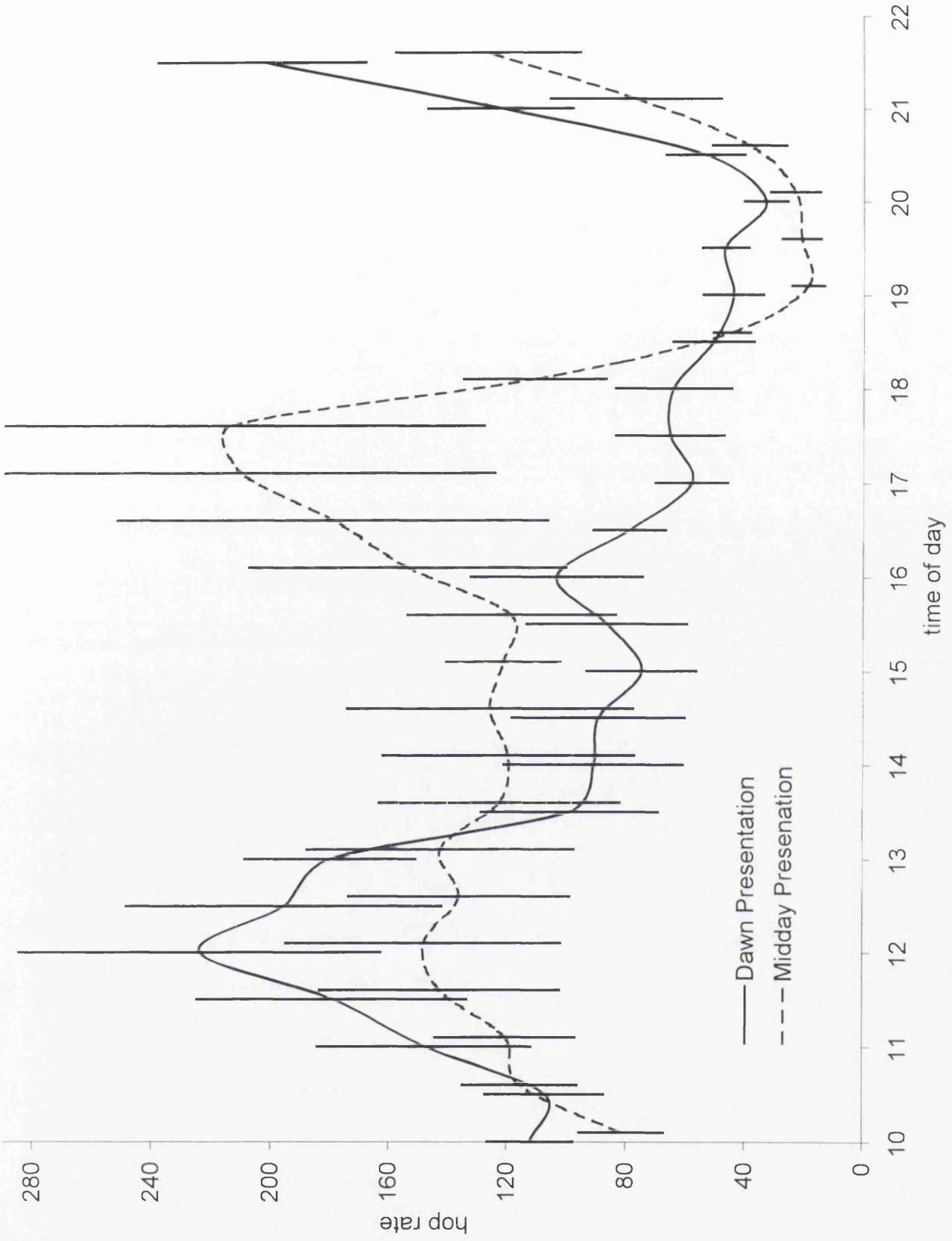


Figure 6. Mean hop rate trajectories (± S.E.) of male zebra finches showing effect due to presentation time; Treatments 1&2 (female presented at dawn) versus 3&4 (female presented at midday). Otherwise data presented as in Figure 2.

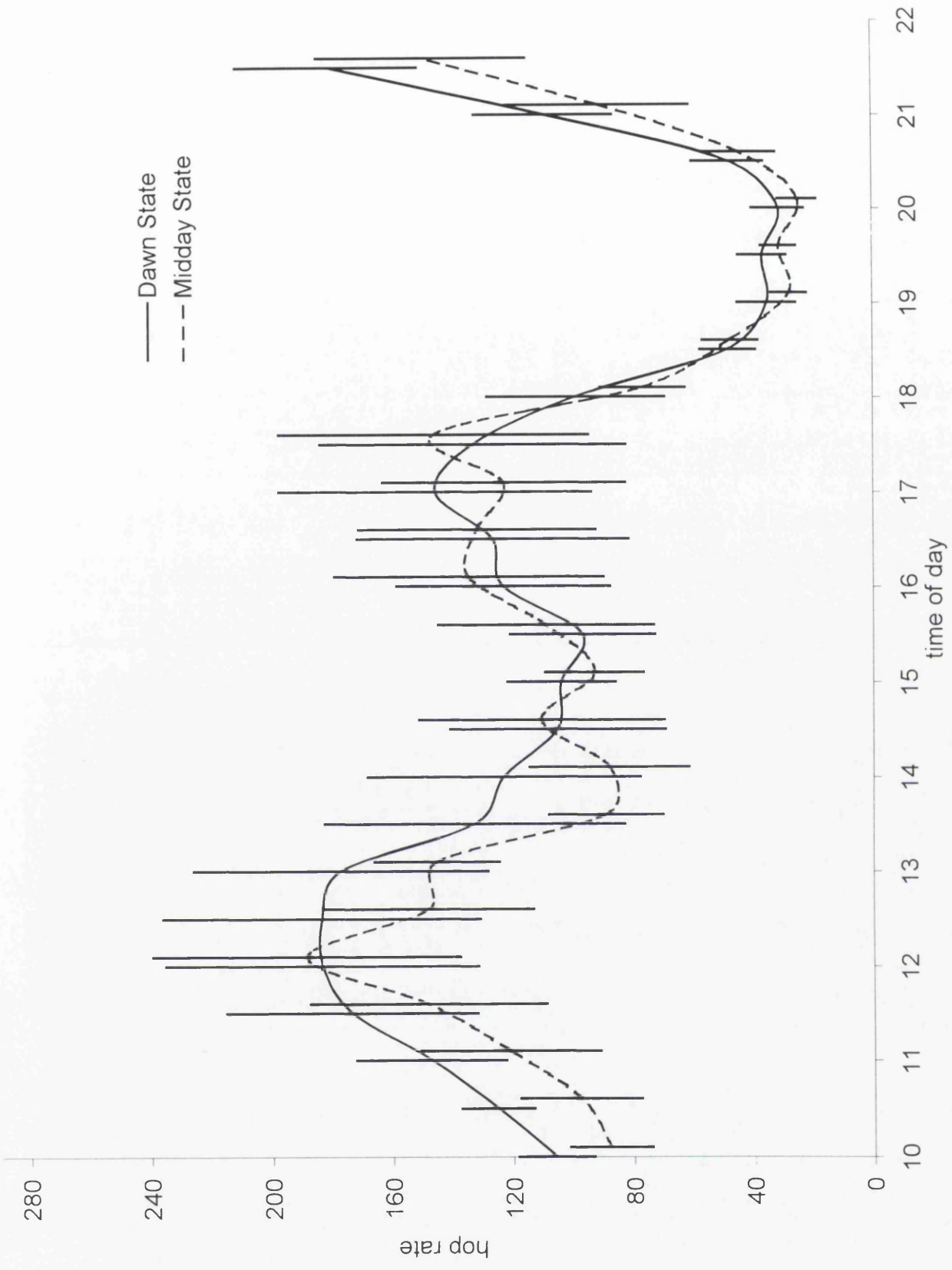


Figure 7. Mean hop rate trajectories (± S.E.) of male zebra finches showing effect due to female state; Treatments 1&3 (female in dawn-state at time of presentation) versus 2&4 (female in midday-state). Otherwise data presented as in Figure 2.

Chapter 6. Arbitrary symmetrical traits of females: their effect on diel patterns of display and mass regulation in male zebra finches

Abstract

Within a fixed time interval, increased time devoted to courtship should decrease the amount of time available for other activities, such as those related to body mass regulation. The attractiveness of prospective mates should influence the amount of time that males devote to courtship. Here there is a potential trade-off between investment in courtship behaviour and maintenance of body mass. We investigated the effects of arbitrary symmetrical and asymmetrical traits of females on the body mass regulation and directed courtship display activity of male zebra finches, Taeniopygia guttata. Female conspecifics, wearing leg bands (two per leg) in either symmetrical or cross-asymmetrical arrangements, were presented singly to the males and allowed to interact through a wire screen. We found no significant differences in male average body mass or daily trajectory of body mass between the leg band treatments, suggesting that leg bands did not influence the trade-off between courtship and feeding activities. Nor were there any effects of leg band treatments on the mean rate or daily pattern of directed display activity, or the proportional use of the most preferred hop perch. There was also no significant effect of female identity (irrespective of band symmetry) on male body mass. However, there was a significant effect of female identity on directed display activity performed by the males. Therefore, while males showed preferences for particular females there was no evidence that arbitrary symmetry influenced males' perception of female attractiveness. This was in contrast to the case of female choice, where symmetry enhances male attractiveness, and indicated that male zebra finches are unlikely to be attentive to arbitrary symmetrical traits in females.

Introduction

Within the boundaries of a given day, an animal must make numerous decisions about the amount of time it allocates to different behaviours, and any time allocated to a given behaviour must detract from the amount of time spent performing an alternative one. Trade-offs between body mass regulation and reproduction have traditionally been viewed as ultimate mechanisms affecting survival and reproductive success (Elmberg & Lundberg, 1991). However, there is also a proximate trade-off between diurnal body mass regulation and investment in reproductive activities. This trade-off could be viewed either in terms of the allocation of biochemical resources to somatic tissue as opposed to gonadal tissue (McManus & Travis, 1998) and/or in terms of time or energy allocated to body mass maintenance and courtship behaviour (Cavallini, 1998; Michener, 1998).

Theoretical models have examined the optimal diurnal patterns of body mass regulation and mate attraction behaviour in birds (Houston & McNamara, 1987; McNamara et al., 1987) and although the underlying theory was developed with singing in mind, it can equally be applied to any courtship behaviour as long as it is both energetically costly and incompatible with foraging (Hutchinson et al., 1993). Previous empirical work (e.g. Mace, 1989; Cuthill & MacDonald, 1990) has focussed only on the temporal pattern of male display behaviour, whereas it is the dynamic interaction between display and mass regulation which is central to the theoretical predictions. Since courtship activity tends to be greatest in the early morning and late evening, an increase in courtship (besides being energetically expensive) may affect both the total amount of time devoted to feeding and the temporal pattern of foraging, hence altering the mean body mass and/or daily mass trajectory.

There has been much recent interest in the role of fluctuating asymmetry in evolutionary processes (Watson & Thornhill, 1994; Møller & Swaddle, 1997). It is hypothesised that fluctuating asymmetry may have an effect on the signalling properties of secondary sexual traits and therefore play an important role in sexual selection (Møller, 1990). There is some empirical evidence to support this (Oakes & Barnard, 1994; Swaddle & Cuthill, 1994a; Morris & Casey, 1998) although a growing body of evidence implies that this relationship may not be generalised (Swaddle, 1999; Swaddle, 2000).

Arbitrary symmetrical traits (specifically coloured plastic leg bands) have been shown to influence female choice in the zebra finch, Taeniopygia guttata (e.g. Swaddle & Cuthill, 1994b; Bennett et al., 1996; Swaddle, 1996; Waas & Wordsworth, 1999; but see Jennions, 1998 for an exception). Similar experiments with bluethroats Luscinia s. svecica have claimed male and female sexual preferences are both influenced by leg band symmetry (Fiske & Amundsen, 1997; Hansen et al., 1999), however both these experiments confounded the symmetry manipulation with other visual cues (see Rohde et al., 1997; Swaddle, 1997 for a discussion).

Swaddle & Cuthill (1994b) found that female zebra finches preferred males wearing symmetrically arranged leg bands. Their experimental design used one-way mirrors which allowed the female to see the stimulus male but not vice versa. This prevented interaction between the male and female but the orientation of the one-way mirror meant that the male stimulus bird may have observed the mirror image of himself which could have influenced his behaviour and hence attractiveness. However, studies where the male could not view himself have also produced the same results, indicating that females preferred males wearing symmetric leg bands (Bennett et al., 1996; Swaddle, 1996; Waas & Wordsworth, 1999). Waas & Wordsworth (1999) did not find a preference for leg band symmetry when males could neither view their own reflection nor the female. When taken in the context of the earlier work by Swaddle & Cuthill (1994b) this suggests that male zebra finches may be sensitive to arbitrary symmetry in an intra-sexual context (c.f. Malyon & Healy, 1994).

Phenomena such as anisogamy and female-biased parental investment are likely to make female mate choice more prevalent and influential than male mate choice (Andersson, 1994). The latter can however be expected to occur to varying extents in monogamous species or in those species in which breeding is highly synchronised (Owens & Thompson, 1994; Kuester & Paul, 1996; Amundsen et al., 1997). The zebra finch fulfils these criteria (Zann, 1996) and male zebra finches have been shown to display quantifiable preferences for specific females (Burley, 1981; Burley et al., 1982; Monaghan et al., 1996).

Having already demonstrated that the presence (chapter 4) or attractiveness/ receptiveness (chapter 5) of female zebra finches influenced the dynamic trade-off between body mass regulation and courtship behaviour in males, we investigated the effect of arbitrary symmetrical traits of stimulus females on this trade-off. This allowed us to assess whether an arbitrary symmetrical trait influenced the attractiveness of female zebra finches in the same way as it has been shown to affect the attractiveness of males of this and other species. As in previous experiments, male preference was assessed by the rate and pattern of hopping directed towards the female stimulus bird . Simultaneously, we recorded the diurnal pattern of body mass regulation of each male. If male zebra finches were attentive to arbitrary symmetrical traits in female conspecifics we predicted that they would show greater directed hopping activity toward symmetrically banded females. Further, if males were attentive to arbitrary symmetry we predicted that there would be an interaction between hopping activity and body mass regulation whereby individuals would trade-off mass (or time spent foraging) for increased time and energy devoted to courtship.

Methods

Subjects & Housing Conditions

Eight male and eight female wild-type zebra finches were used in this study. To ensure that the two sexes had no prior knowledge of one another we obtained the sexes from separate breeders. None of the males had bred for at least one year prior to the start of this study, and none of the females had bred at all. All birds were between 2-4 years old. When not being used in the experiment, all birds were housed in single sex cages (99 x 29 x 39 cm high) located in the same room. The cages were arranged such that the two sexes were visually, but not acoustically, isolated from one another. Throughout the experiment (and for 3 months beforehand) all birds were maintained on a 14:10 h light/dark photoperiod which included one hour of artificial dawn and one hour of artificial dusk. During these periods the light intensity increased (at dawn) and decreased (at dusk) by means of the sequential switching of a complex lighting array. Temperature within the room was maintained at mean $\pm$ S.D. = 19.49 $\pm$ 2.10°C, n=864, before and during the experiment. Fresh drinking and bathing water, mixed seed, oyster shell grit and cuttlebone were provided *ad libitum*. In addition, all birds were provided on alternate days with freshly cut grass except when occupying the experimental apparatus. All daily husbandry procedures were carried out during the middle hour of the day when activity amongst the birds was relatively low, in order to minimise and standardise disturbance.

Experimental Cages

The eight experimental cages (147 x 29 x 39 cm high; Figure 1) consisted of three equally sized sections: food, courtship and stimulus. In each cage a male had access to the food and courtship sections and a female was placed in the stimulus section. The food and courtship sections were separated by a pair of opaque baffles which permitted the male bird to move freely between these two sections but prevented the male from viewing the female whilst he was in the feeding section. The courtship and stimulus sections were separated by a stainless steel mesh composed of vertical wires of 1mm diameter set at 15 mm centres with a single horizontal wire set above the birds' line of view. The food section contained a water dish and a food hopper that was accessible from a microswitch perch, however, the male birds did not always choose to feed from

here. The courtship section contained a set of four microswitch perches arranged in a square with opposite pairs set 12 cm apart and 14 cm above the cage floor, hence these perches varied in their distance and orientation to the female. The perches were made from 5 cm lengths of 1.3 cm diameter dowel. These perches recorded the temporal and spatial pattern of display hopping activity. The wire mesh which separated the courtship and stimulus sections was 4 cm from the nearest display perch. The time of day and duration of hopping activity on all microswitch perches was recorded remotely on a PC. Male zebra finches hop repeatedly between adjacent perches or perform 'head-tail twists' as part of their courtship display (Morris, 1954; Zann, 1996), and so this apparatus provided a reliable method of automatically recording sexual display activity. The set of four display perches was mounted on an electronic balance (Sartorius PT-610) which read to 0.1 g accuracy. The balances from each of the 8 experimental cages were connected to the PC, allowing automated recording of body mass of each male. Computer software automatically detected and corrected for drift in the output of the balances.

The stimulus (i.e. female) section was equipped with two perches positioned 7 cm and 42 cm from the dividing wire mesh and set at the same height as the male display perches. The back wall of the stimulus section was white so that the male always viewed the stimulus female against a white background. The stimulus section was bedded with composted wood bark, but the floor of the food and courtship sections was plain hardboard, to prevent excessive debris being deposited by the males on either the perch mechanisms or the balances.

Band Combinations

We used four of the band combinations (two symmetrical and two cross-asymmetrical) used by previous studies (e.g. Swaddle & Cuthill, 1994b; Waas & Wordsworth, 1999). It has been shown that male zebra finches prefer black banded females and dislike those wearing light green bands (Burley, 1981; Burley et al., 1982). We therefore used these two colours, light green (G) and black (B), due to their contrast in both colour and male preference. All females wore four bands in total, with one of each colour on each leg. The consequence of this was that the total colour per female, and also the total colour on each leg, was constant for all treatments, with only relative symmetry being varied. The

four treatments in order 1 to 4 were GB:GB, BG:BG, BG:GB and GB:BG (given as 'top left /bottom left: top right/ bottom right). Treatments 1 and 2 are symmetrical and treatments 3 and 4 are asymmetrical. Leg bands were supplied by A.C. Hughes Ltd. (Middlesex, U.K.)

Experimental Procedure

Each female was presented to each male on four occasions and on each occasion the female wore a different leg band arrangement. The experiment was divided into four blocks of eight days. During each block the experimental males were presented with each of the eight females for one day. The order in which the females appeared and the band arrangements they wore were randomised except that the same band arrangement (albeit on a different female) was not presented to a given male bird on consecutive days. Hence males could not predict which female or leg band treatment they would experience on any day of the experiment. In between the four experimental blocks all birds were returned to single sex cages (which were in visual isolation) for two full days. During trials, female birds were moved between cages during the last six minutes of the day during which time the light level had fallen to near darkness and the male birds were virtually inactive. This ensured that the correct female was in place at first light the next day and that no disruption occurred at dawn.

Statistical Analyses

Repeated-measures ANOVA tests were used to determine whether the different leg band combinations had any significant effects on male mass regulation or hopping activity patterns. Display perch activity and mass data were condensed into total perch movements and mean mass per half hour interval respectively. Only data for the period of daylight were used in the analyses, giving 28 data points per parameter for each bird on each day. The repeated measures model contained two within-subject variables, namely Treatment (4 levels) and Time of Day (28 levels). For each male, body masses and hopping rates at each time of day were averaged across the 8 stimulus females presented for a given treatment. We also examined the effect of the individual females' identity (irrespective of leg banding symmetry) on male mass regulation and activity patterns. Data were again condensed into 28 daily intervals, and the repeated-measures

model contained two within-subject variables, Female (8 levels) and Time (28 levels). The attentiveness of males towards females was ranked in terms of total hops performed by each male to a particular female. The level of agreement between the attentiveness hierarchies of the males was found by calculating Kendall's coefficient of concordance (*W*).

Differences in the proportional use of hop perch 1 (see Figure 1 to identify this perch), were considered among treatments and also in terms of female identity. Perch 1 was chosen for detailed analysis because it was, on average, the most used perch and offered an orientation commonly adopted during courtship (see later). This perch was considered in isolation in order to preserve the independence of data points. The proportional use was measured by calculating the (arcsin-transformed) proportion of the total hops performed on display perch 1 per half hour interval.

The repeated measures ANOVA statistics presented here are either univariate or multivariate as appropriate depending on the outcome of Mauchly's test of sphericity (Zar, 1996). Two-tailed tests of probability were used throughout.

Results

There was a significant effect of time on hop rate ($F_{27,189} = 13.60$, $P < 0.0001$) and on body mass ($F_{27,189} = 72.66$, $P < 0.0001$), which simply confirmed that that the temporal pattern of both hopping activity and body mass fluctuated throughout the day. However, there were no significant effects of leg band combinations on either male mass or hopping activity, (mass $F_{3,5} = 1.37$, $P = 0.35$, Figure 2; hop rate $F_{3,21} = 0.32$, $P = 0.81$, Figure 3). The treatment-by-time interaction term relating to hop rate was also not significant ($F_{81,567} = 0.71$, $P = 0.97$) and by tending towards unity it suggested that the daily trajectory (i.e. pattern) of hopping activity was highly consistent among leg band treatments (Figure 3). The treatment-by-time interaction term for body mass was also not significant ($F_{81,567} = 1.06$, $P = 0.35$), indicating that body mass trajectories did not differ among leg band treatments.

When effects of leg band combinations were ignored, and data were analysed only in terms of effects due to female identity, there was no significant effect of female identity on body mass ($F_{7,1} = 3.36$, $P = 0.40$; Figure 4). However, there was a significant effect of female on hopping activity ($F_{7,1} = 293.6$, $P = 0.045$; Figure 5). The latter result indicates that males displayed more in front of certain females than others. Multivariate statistics are given in both cases, due to assumptions about sphericity being broken, and therefore female-by-time interaction terms are not available (Zar, 1996) . When male attentiveness towards particular females was ranked in terms of total hops performed, there was a significant level of agreement between males (Kendall's $W = 0.33$, $P = 0.009$).

Display perch 1 was the most preferred perch, and display perch 3 the least preferred (Figure 6). There were no significant differences in the proportional use of display perch 1 among leg band treatments ($F_{3,5} = 0.45$, $P = 0.72$) or due to female identity ($F_{7,1} = 0.81$, $P = 0.70$). There was however a significant difference in proportional use of perch 1 across the day, with it being most heavily used in the early morning and least used at the very end of the day ($F_{27,189} = 2.56$, $P<0.0001$). The treatment-by-time interaction term was available only for the effect due to leg bands and there was no significant effect ($F_{81,567} = 0.70$, $P = 0.98$).

Discussion

Male zebra finches do not appear to respond (in terms of courtship activity or mass regulation strategy) to the symmetry of a female's coloured leg bands, in contrast to the situation where the sex roles are reversed (Swaddle & Cuthill, 1994b; Bennett et al., 1996; Swaddle, 1996; Waas & Wordsworth, 1999). In the present study, no significant effects of leg band treatment were found on mean hopping activity rate, mean body mass of males, their daily pattern of hopping activity, gain in body mass or in their proportional usage of display perch 1. This is despite the fact that the males were clearly engaging in frequent bouts of courtship as is evident from the intense use of the hop perches in the early morning and evening, when courtship normally occurs (Mace, 1987; McNamara et al.1987; Møller, 1991).

One way of interpreting the absence of a relationship between symmetry treatments and display activity may be that the design of the experimental apparatus, in particular the absence of a choice chamber design, was not conducive to promoting differential male effort in response to the various stimuli. Arguably, when presented with only one stimulus at a time the male bird may have opted to display maximally and/or equally to all stimuli, largely irrespective of phenotype. However, the aim of this experiment was to test the effects of arbitrary symmetrical traits on the dynamic interaction between body mass regulation and courtship behaviour. A choice chamber test would have ruled out any investigation of mass dynamics in general and specifically whether investment in body mass was traded-off against display effort. Therefore, the chosen experimental design was the more appropriate to test our specific hypothesis, particularly because it has already been shown to be capable of evoking and detecting this trade-off (chapters 4, 5 & 7).

To test the idea that because rejecting an unfavourable stimulus meant refraining from courtship, males would opportunistically display maximally and/or equally to all stimuli, we removed any effects due to banding from the analysis and examined whether males exhibited any consistent differences in attentiveness towards the individual female birds. Our approach was appropriate as each female appeared before each male on four occasions, each time wearing a different arrangement of leg bands. The order in which the females were presented was randomised to minimise any

potential systematic effects that may otherwise have arisen. Therefore, the effect of leg banding in this analysis is balanced and may be disregarded without risk of bias. Females will naturally vary in attractiveness, hence this form of analysis could reveal the influence of natural variance in female attractiveness on male courtship activity and male body mass regulation. When the data were analysed purely in terms of female identity (leg band treatment being ignored), there was still no effect on body mass. However, males did show significant and consistent (between-male) variation in their hopping activity in front of particular females. This refutes the notion that the male birds, through lack of real time choice, were displaying (or directing hopping) maximally to all stimulus females irrespective of artificial or natural phenotypic qualities. Indeed, there was a significantly consistent hierarchy of attentiveness towards individual females by the experimental males, as indicated by hop rate (when leg band symmetry was removed from the analysis).

The absence of any change in mean body mass or the daily body mass trajectory despite significant variation in attentiveness to individual females is somewhat anomalous. Theory predicts that the cost of increased courtship will be reflected in decreased body mass (Hutchinson et al., 1993). However, zebra finches spend approximately half of the day inactive (Dall & Witter, 1998) so there is considerable scope for balancing increased display with strategically increased foraging at the expense of time devoted to inactivity. An examination of the length or frequency of feeding bouts would confirm this, but although the experimental apparatus employed a microswitch feeding perch, it was, unfortunately, impossible to ensure that the male birds always perched there to feed. Certain individuals opted to land on the rim of the feeding dish despite efforts to make this an unattractive platform. Consequently, the idea that males increased their courtship activity but maintained their body mass by increasing their rate of food intake while feeding is difficult to test here (c.f. Dall & Witter, 1998, where food deprived zebra finches maintained their body mass but traded off energetic behaviours in favour of increased inactivity).

Although contrary to theoretical predictions, the lack of relationship between investment in courtship activity and change in body mass regulation strategies is supported by some observational data in other taxa. For example, there is no relationship between investment of time in costly mate attraction behaviour and body

mass in a wolf spider Hygrolycosa rubrofasciata (Kotiaho et al., 1999). Similarly, male body mass does not vary with rates of intrusions on to neighbouring territories to gain extra-pair copulations in hooded warblers Wilsonia citrina (Stutchbury, 1998). However, several other researchers have provided correlational evidence of a trade-off between body mass reserves and amount of mate solicitation behaviour, e.g., in Richardson's ground squirrels Spermophilus richardsonii (Michener, 1998), red foxes Vulpes vulpes (Cavallini, 1998) and female black grouse Tetrao tetrix (Rintamaki et al., 1995).

Given the very marked differences in adult male and female zebra finch plumage (Morris, 1954; Zann, 1996) it is perhaps unsurprising that females do and males do not exhibit intrinsic preferences for arbitrary symmetrical traits in the opposite sex. The male plumage (as perceived by a human observer, but see Bennett et al., 1996 and Hunt et al., 1997) is greatly predisposed to reveal asymmetries due to the prevalence of bilateral colourful and contrasting markings. In sharp contrast the female birds' most prominent features (albeit from a human perspective) are her cheek bars and beak, and here there is limited scope to provide mate choice cues pertaining to the degree of plumage symmetry. It seems more likely that other signals such as bill colour (Price & Burley, 1994), vocalisations (Eens & Pinxten, 1998) or age and plumage brightness (Hill, 1993) will be used by male zebra finches in assessing potential mates.

Although males zebra finches are not attentive to arbitrary symmetry in females, there is a strong indication that they may use it as a signal in an intra-sexual context (see Swaddle & Cuthill, 1994b and Waas & Wordsworth, 1999). There is evidence in the literature to indicate that male zebra finches provide symmetry cues via their plumage (Swaddle & Cuthill, 1994a) and females are undoubtedly attentive to artificial symmetry cues in males. Given this evidence, but in the absence of a study which measures females responses to pre-existing natural fluctuating asymmetries, it would appear that at least one way in which male zebra finches signal their fitness is by means of the level of plumage or corporeal symmetry. Assuming that such signals exist, they could potentially be exploited by other males in male-male competitive encounters and in the establishment of hierarchies (Malyon & Healy, 1994). However, Swaddle (1996) found that the same leg band treatments as used in this study do not influence male dominance hierarchies when males are housed in single-sex or mixed-sex groups.

Perch 1, which was perpendicular to the female and nearest the cage front (Figure 1), was generally the preferred perch with the male typically spending around 40% of his time there. The proportional use of this perch did not vary with leg band treatment or the identity of the female, but it did vary with time of day. The use of this perch declined throughout the day, most noticeably at dusk, while the use of perch 2 (nearest the female) increased sharply at this time (Figure 6). By alighting on perch 1, the male afforded the female a view of one or other of his flanks. This posture is typically adopted during the initial 'static' phase of the courtship process, when the male bird typically engages in serial lateral presentations to the female (Morris, 1954; Workman & Andrew, 1986). His posture during these displays would serve to display prominently his sex-specific markings (e.g. chestnut cheek patches, white spotted chestnut flanks, finely barred black and white throat and chest with large black margin to lower breast) and presumably also provide an opportunity for females to assess bilateral symmetry (c.f. Morris & Casey, 1998; but see Shettleworth, 1999).

The extreme difference in the use of perches 1 and 3 (Figure 6), either of which would afford a side-on posture, is most likely due to perch 1 being in closer proximity to the wire cage front from where the male had a better view of his surroundings and experienced marginally higher ambient light levels. In addition, Workman & Andrew (1986) found a significant tendency for courting male zebra finches to display their right hand flanks (and thus view the female using their right eye). Perch 1 is more suited to presenting the right flank as it allows the male to view the other three perches and to access them without first turning. Presenting the right flank from perch 3 would prevent this, and would entail facing the back of the cage, in slightly less intense light, both of which are factors that may hamper vigilance.

In conclusion, this study is the first to experimentally test the attentiveness of male birds to arbitrary symmetrical traits in females in a manner which does not confound effects due to symmetry with other factors. Although there is evidence that male zebra finches may be attentive to such artificial traits in males, they are not attentive to arbitrary symmetry in females. However, there is an effect of natural variation in female appearance on male display activity, but this increased investment in display does not apparently influence body mass regulation strategies of males.

References

Amundsen, T., Forsgren, E. & Hansen, L. T. T. 1997. On the function of female ornaments: male bluethroats prefer colourful females. Proceedings of the Royal Society of London Series B, 264, 1579-1586.

Andersson, M. 1994. Sexual Selection. Princeton University Press: Princeton.

Bennett, A. T. D., Cuthill, I. C., Partridge, J. C. & Maier, E. J. 1996. Ultraviolet vision and mate choice in zebra finches. Nature, 380, 433-434.

Burley, N. 1981. Sex Ratio Manipulation and Selection for Attractiveness. Science, 211, 721-722.

Burley, N., Krantzberg, G. & Radman, P. 1982. Influence of colour-banding on the conspecific preferences of zebra finches. Animal Behaviour, 30, 444-455.

Cavallini, P. 1998. Differential investment in mating by red foxes. Journal of Mammalogy, 79, 215-221.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Eens, M. & Pinxten, R. 1998. Female song for mate attraction: an overlooked phenomenon. Trends in Ecology and Evolution, 13, 322-323.

Elmberg, J. and Lundberg, P. 1991. Intraspecific variation in calling, time allocation and energy reserves in breeding male common frogs Rana temporaria. Annales Zoologici Fennici, 28, 23-29.

Fiske, P. & Amundsen, T. 1997. Female bluethroats prefer males with symmetric colour bands. Animal Behaviour, 54, 81-87.

Hansen, L. T. T., Amundsen, T. & Forsgren, E. 1999. Symmetry: attractive not only to females. Proceedings of the Royal Society of London Series B, 266, 1235-1240.

Hill, G. E. 1993. Male mate choice and the evolution of female plumage coloration in the house finch. Evolution, 47, 1515-1525.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Hunt, S., Cuthill, I. C., Swaddle, J. P. & Bennett, A. T. D. 1997. Ultraviolet vision and band-colour preferences in female zebra finches Taeniopygia guttata. Animal Behaviour, 54, 1383-1392.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Jennions, M. D. 1998. The effect of leg band symmetry on female-male association in zebra finches. Animal Behaviour. 55, 61-67.

Kotiaho, J.S., Alatalo, R.V., Mappes, J. and Parri, S. 1999. Sexual signalling and viability in a wolf spider (Hygrolycosa rubrofasciata): measurements under laboratory and field conditions. Behavioural Ecology and Sociobiology, 46, 123-128.

Kuester, J. & Paul, A. 1996. Female-female competition and male mate choice in Barbary macaques (Macaca sylvanus). Behaviour, 133, 763-790.

Mace, R. 1987. The dawn chorus in the great tit Parus major is directly related to female fertility. Nature, 330, 745-746.

Mace, R. 1989. Great tits choose between food and proximity to a mate: the effect of time of day. Behavioural Ecology and Sociobiology, 24, 285-290.

Malyon, C. & Healy, S. 1994. Fluctuating asymmetry in antlers of fallow deer, Dama dama, indicates dominance. Animal Behaviour, 48, 248-250.

McManus, M.G. and Travis, J. 1998. Effects of temperature and salinity on the life history of the sailfin molly (Pisces: Poeciliidae): lipid storage and reproductive allocation. Oecologia, 114, 317-325.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Michener, G.R. 1998. Sexual differences in reproductive effort of Richardson's ground squirrels. Journal of Mammalogy, 79, 1-19.

Møller, A. P. 1990, Fluctuating asymmetry in Male Sexual Ornaments May Reliably Reveal Male Quality. Animal Behaviour, 40, 1185-1187.

Møller, A. P. 1991. Why mated songbirds sing so much: mate guarding and male announcement of mate fertility status. American Naturalist, 138, 994-1014.

Møller, A. P. & Swaddle, J. P. 1997. Asymmetry, developmental stability and evolution. Oxford: Oxford University Press.

Monaghan, P., Metcalfe, N. B. & Houston, D. C. 1996. Male finches selectively pair with fecund females. Proceedings of the Royal Society of London Series B, 263, 1183-1186.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Morris, M. R. & Casey, K. 1998. Female swordtail fish prefer symmetrical sexual signal. Animal Behaviour, 55, 33-39.

Oakes, E. J. & Barnard, P. 1994. Fluctuating asymmetry and mate choice in paradise whydahs, Vidua paradisaea: an experimental manipulation. Animal Behaviour, 48, 937-943.

Owens, I. P. F. & Thompson, D. B. A. 1994. Sex differences, sex ratios and sex roles. Proceedings of the Royal Society of London Series B, 258, 93-99.

Price, D. K. & Burley, N. T. 1994. Constraints on the evolution of attractive traits: selection in male and female zebra finches. The American Naturalist, 144, 908-934.

Rintamaki, P. T., Alatalo, R. V., Höglund, J. and Lundberg, A. 1995. Mate sampling behavior of black grouse females (Tetrao tetrix). Behavioural Ecology and Sociobiology, 37, 209-215.

Rohde, P. A., Amundsen, T. & Fiske, P. 1997. Fluctuating asymmetry, mate choice and experimental designs. Animal Behaviour, 54, 1030-1033.

Shettleworth, S. J. 1999. Female mate choice in swordtails and mollies: symmetry assessment or Weber's law? Animal Behaviour, 58, 1139-1142.

Stutchbury, B. J. M. 1998. Extra-pair mating effort of male hooded warblers, Wilsonia citrina. Animal Behaviour, 55, 553-561.

Swaddle, J. P. 1996. Reproductive success and symmetry in zebra finches. Animal Behaviour, 51, 203-210.

Swaddle, J. P. 1997. Experimental design and the signalling properties of fluctuating asymmetry. Animal Behaviour, 54, 1034-1037.

Swaddle, J. P. 1999. Limits to length asymmetry detection in starlings: implications for biological signalling. Proceedings of the Royal Society of London Series B, 266, 1299-1303.

Swaddle, J. P. 2000. Is fluctuating asymmetry a visual signal? In: Animal Signals-Adaptive Significance of Signalling and Signal Design in Animal Communication (Ed. by Y. Espmark, T. Amundsen & G. Rosenqvist). Proceedings Fifth International Kongsvoll Symposium, September 15-20, 1998. Trondheim, Norway: Tapir Publishers. In press.

Swaddle, J. P. & Cuthill, I. C. 1994a. Female zebra finches prefer males with symmetric chest plumage. Proceedings of the Royal Society of London Series B, 258, 267-271.

Swaddle, J. P. & Cuthill, I. C. 1994b. Preference for symmetric males by female zebra finches. Nature, 367, 165-166.

Waas, J. R. & Wordsworth, A. F. 1999. Female zebra finches prefer symmetrically banded males, but only during interactive mate choice test. Animal Behaviour, 57, 1113-1119.

Watson, P. J. & Thornhill, R. 1994. Fluctuating asymmetry and sexual selection. Trends in Ecology and Evolution, 9, 21-25.

Workman, L. & Andrew, R. J. 1986. Asymmetries of eye use in birds. Animal Behaviour, 34, 1582-1584.

Zann, R. A. 1996. The Zebra Finch: A Synthesis of Field and Laboratory Studies. New York: Academic Press.

Zar, J. H. 1996. Biostatistical Analyses. 3rd edn. New Jersey: Prentice Hall International Inc.

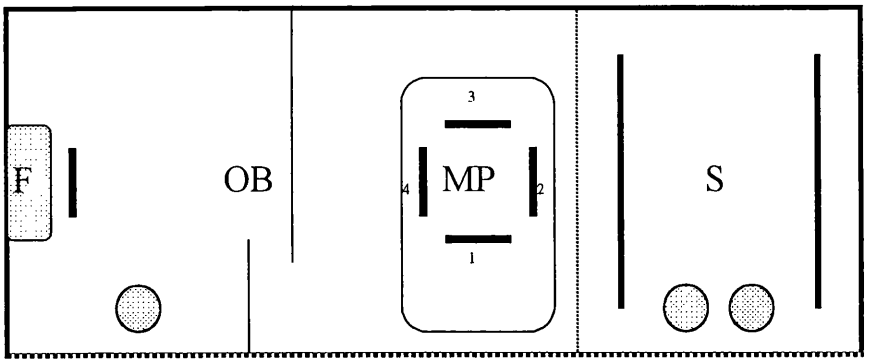


Figure 1. The experimental cage viewed from above. F = feed hopper, OB = opaque baffles, MP = microswitch perches mounted on electronic balance, S = stimulus cage. Heavy bars represent perches, dotted lines represent cage front and wire mesh separating courtship and stimulus compartments. Shaded circles are food and water dishes.

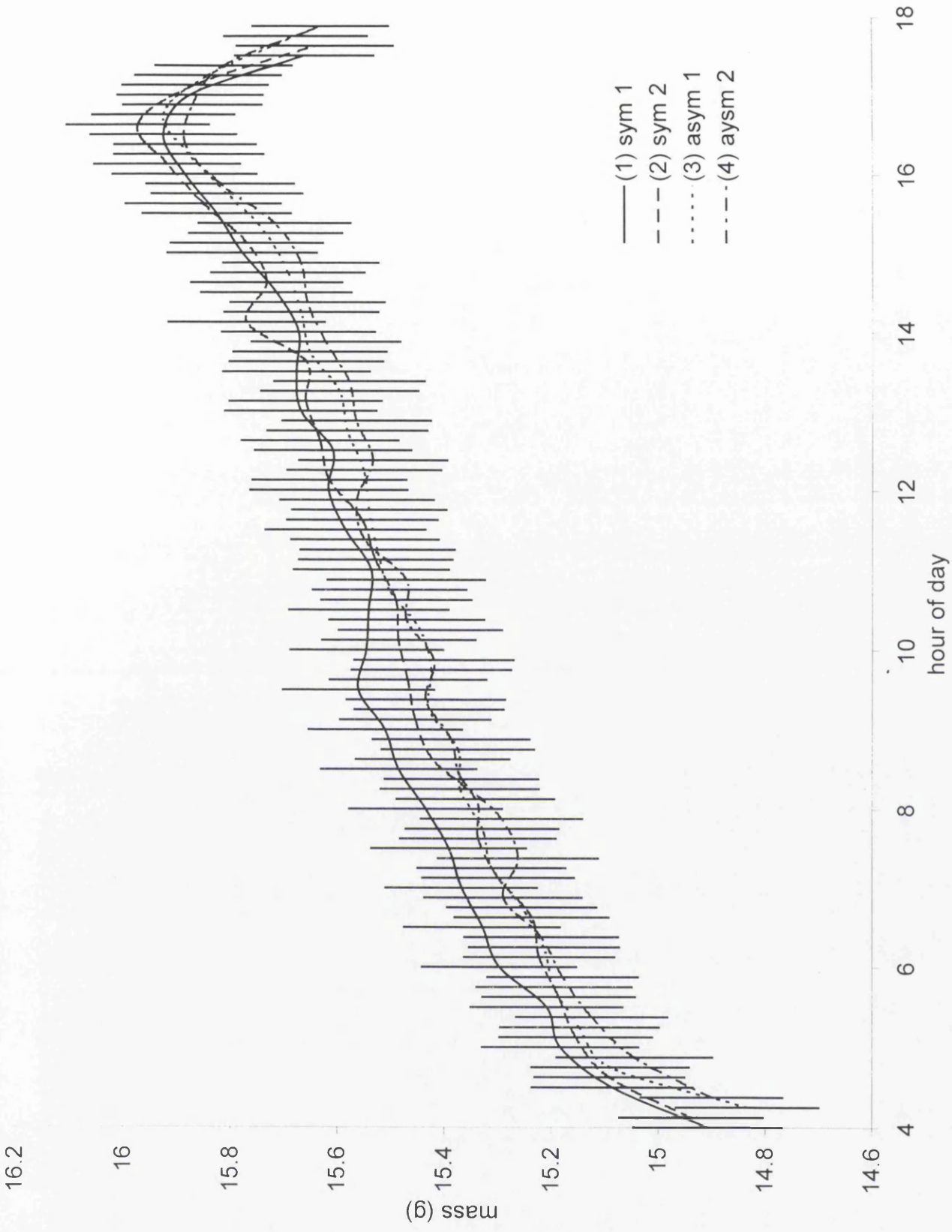


Figure 2. Mean body mass trajectories (± S.E.) of male zebra finches presented with a female wearing one of the four leg band treatments. X-axis values are staggered among treatments by 7.5 minutes for clarity (to prevent error bars from overlapping). Treatment 1, GB:GB; Treatment 2, BG:BG; Treatment 3, BG:GB and Treatment 4, GB:BG (G is light green, B is Black. Order shown as top left/bottom left: top right/bottom right).

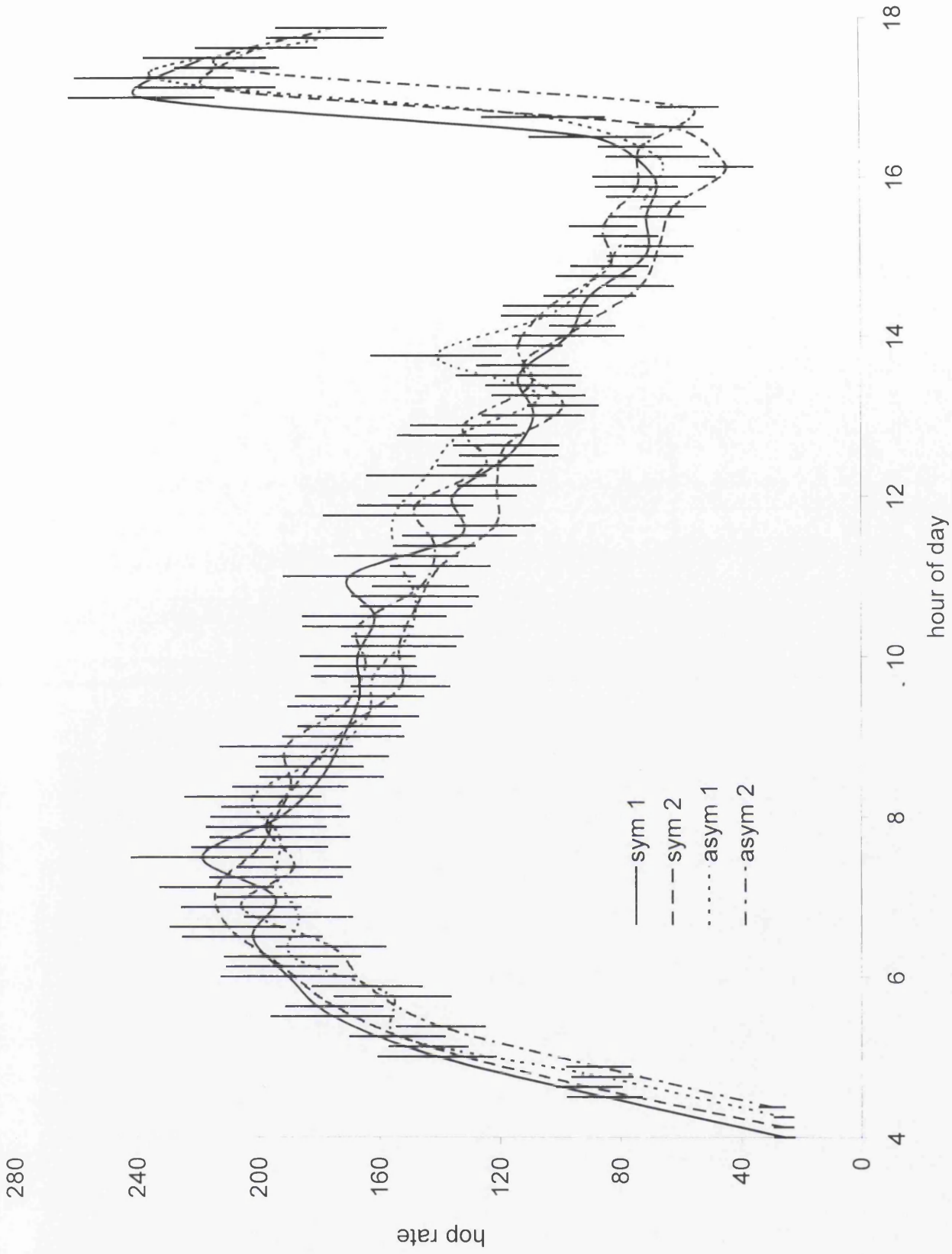


Figure 3. Mean hop rate trajectories (± S.E.) of male zebra finches presented with a female wearing one of the four leg band treatments. Data presented as in Figure 2.

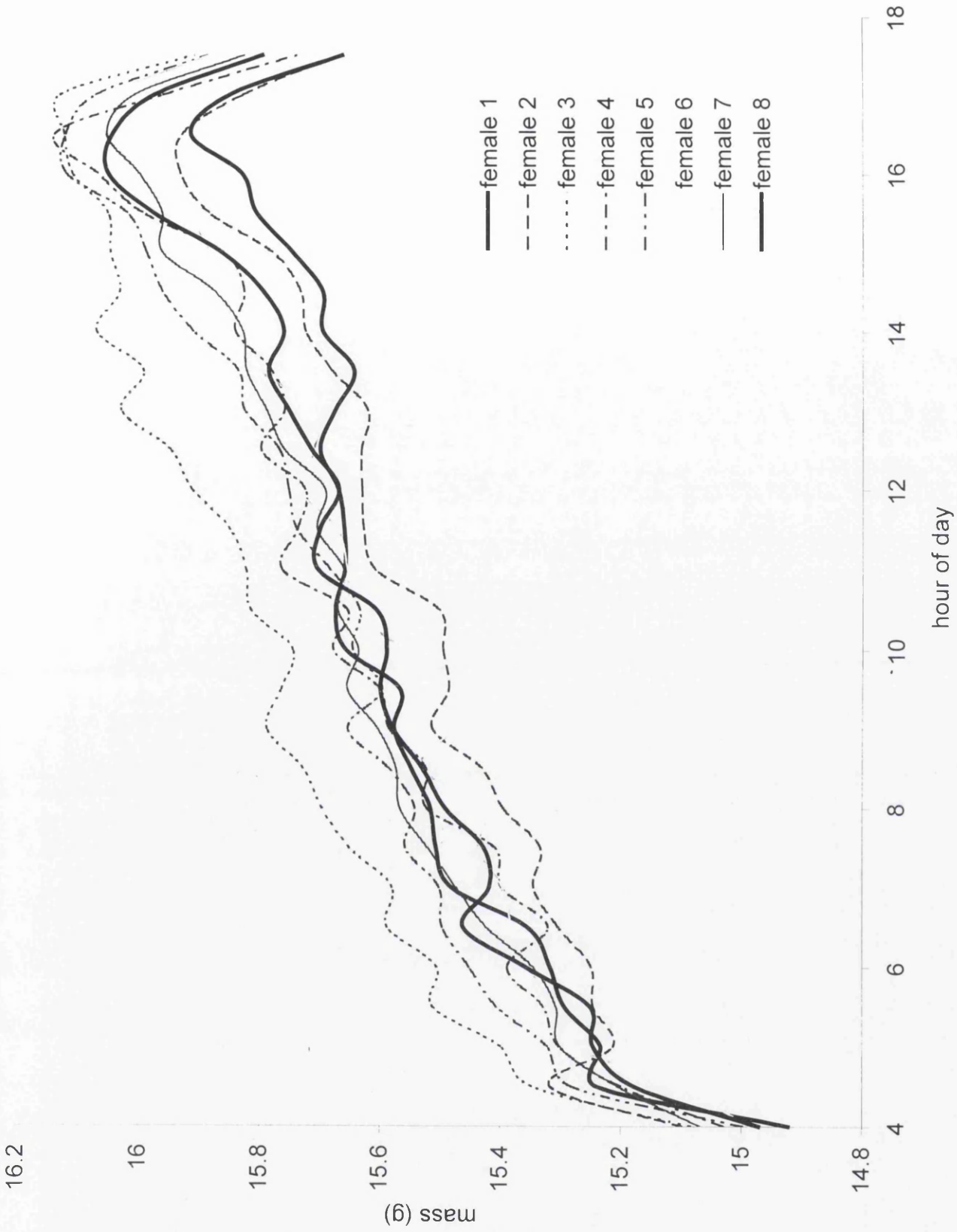


Figure 4. Mean body mass trajectories of male zebra finches when presented with each of the eight stimulus females (banding ignored). Error bars omitted for clarity.

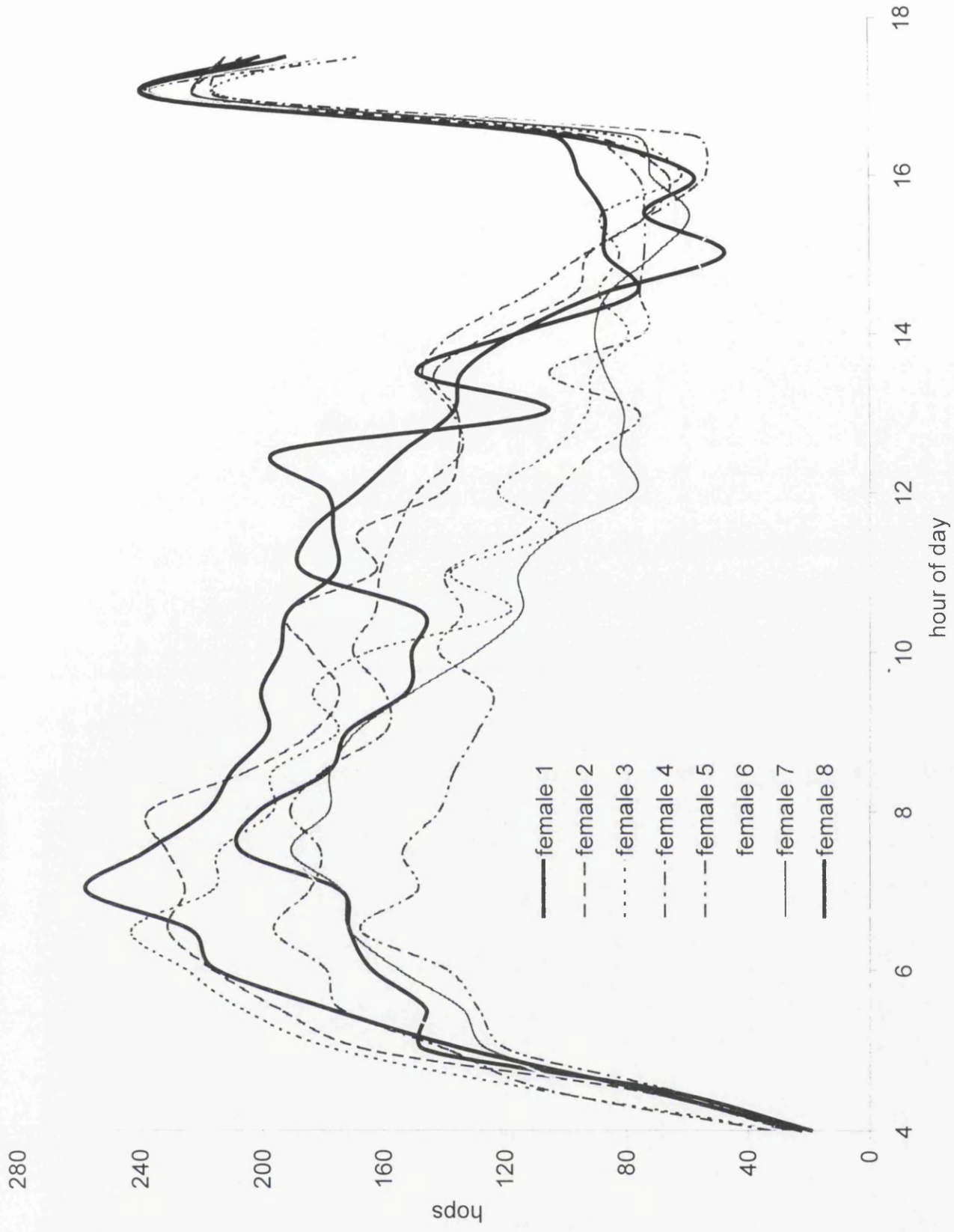


Figure 5. Mean hop rate trajectories of male zebra finches when presented with each of the eight stimulus females (banding ignored). Error bars omitted for clarity.

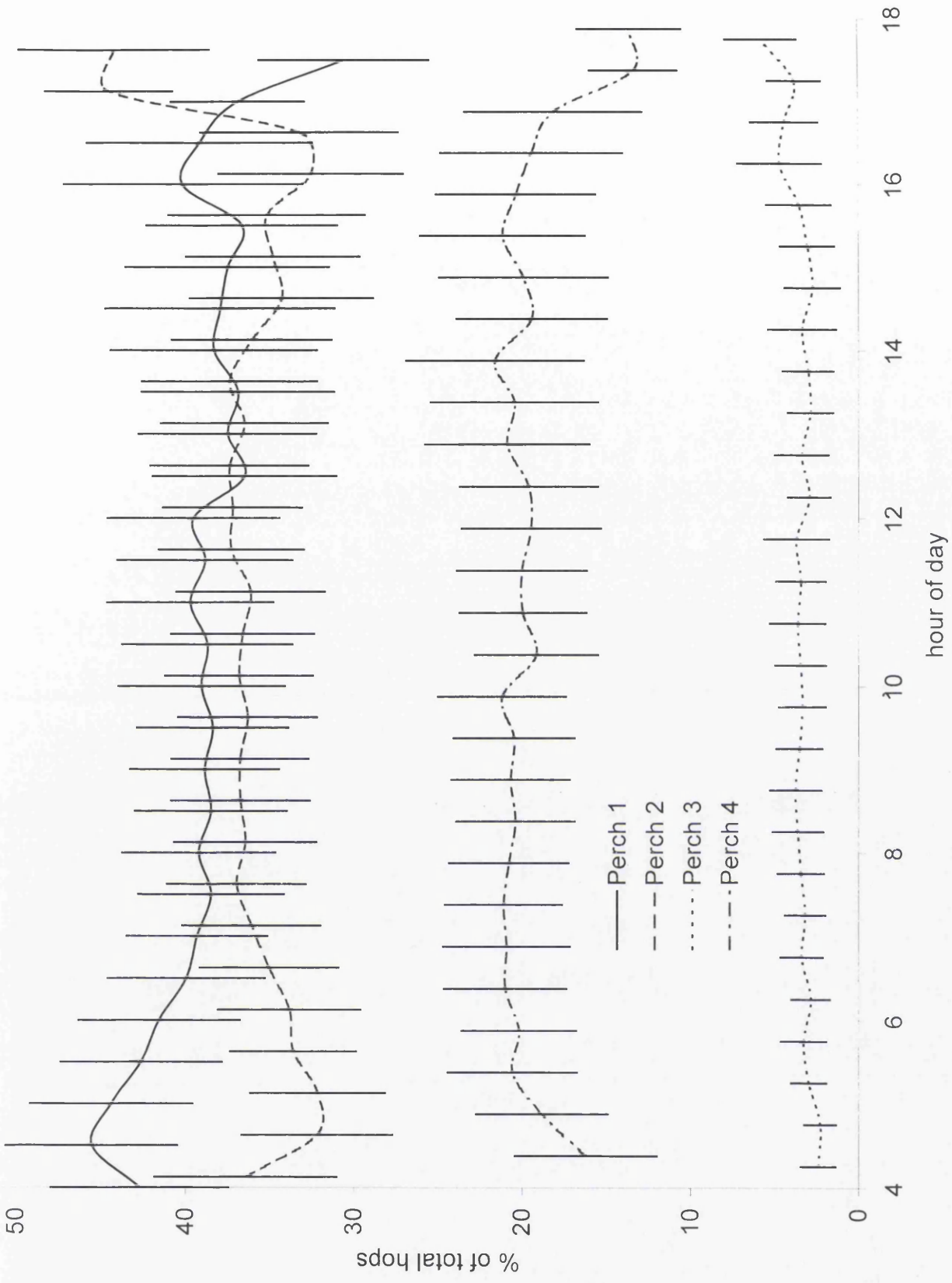


Figure 6. Mean proportion of the total hops performed on each individual hop perch. Data presented as in Figure 2.

Chapter 7. The effects of food restriction on diel patterns of body mass and display activity in male zebra finches

Abstract

In general, foraging behaviour is incompatible with courtship behaviour, therefore, the decision to engage in courtship creates a trade-off against body mass regulation. Stochastic dynamic programming models predict how patterns of change in the environment, for example variable food availability, will influence that trade-off, and they examine the role of those patterns in forming and modulating the peaks of display behaviour at dawn and dusk common in many small passerines. We investigated the effects of food restrictions on the temporal patterns of body mass regulation and directed hopping display of male zebra finches (Taeniopygia guttata). Our results are contrary to theoretical predictions and a number of similar empirical studies. We found significant effects of variable foraging success on the trajectory of body mass across the day, however, we found no effects on mean body mass, mean directed hopping activity or hopping activity trajectories. There was in fact a non-significant tendency for hopping activity to be higher or highest specifically during periods of the day when food was unavailable. A cost of courtship is time lost from foraging and an anomaly is that when foraging is not possible these costs become zero. During food restriction therefore, courtship is relatively less costly but the risk of starvation is greater - however male zebra finches appeared to overlook this increased risk. Males responded quickly to changes in the temporal pattern of food availability and their responses remained consistent throughout the duration of each treatment. This implied that the males' strategy was to base the state of their reserves on current as opposed to historical information. Given the plasticity with which somatic reserves were regulated, male zebra finches would appear to be well equipped to adopt that strategy.

Introduction

Recent stochastic dynamic programming models of the diurnal organisation of courtship and foraging in small passerines (Houston & McNamara, 1987; McNamara et al., 1987; Hutchinson et al., 1993) have explored the optimal routines which arise from the basic assumptions that courtship is both costly and mutually exclusive with foraging activity. These models predict the ways in which the dynamic interaction between foraging and courtship behaviour will vary with circadian patterns in the environment (for example ambient temperature and food availability) to allow the individual to maximise its gain from courtship while simultaneously avoiding starvation. Specifically the models predict daily routines of courtship with peaks at dawn and dusk, similar to those of many species of songbird, and that variability in foraging success will reduce display levels and increase body mass. A key assumption of these models is the existence of a crossover point in food reserves below which it is optimal to forage and above which it is optimal to display. An increased probability of foraging being interrupted increases the level of this crossover point, as an insurance against arriving at dusk with insufficient reserves, which therefore leads to the reduced display effort and elevated body mass as described.

Several empirical studies have tested certain predictions of these models. In greenfinches (Carduelis chloris L.), predictable foraging success led to reduced body masses and vice versa (Ekman & Hake, 1990). In great tits (Parus major), delaying the start of feeding led to increased dawn masses, while unpredictable food availability led to greater daily mass gains (Bednekoff & Krebs, 1995). In blackbirds (Turdus merula), food supplementation has been shown to increase the duration and magnitude of courtship song in paired males (Cuthill & MacDonald, 1990) while food deprivation has been shown to induce increased body masses in European starlings (Sturnus vulgaris) (Witter et al, 1995; Witter & Swaddle, 1997). A study by Dall & Witter (1998) on zebra finches (Taeniopygia guttata) found no variation in body mass regulation before and after a period of unpredictable foraging interruptions, but did find a reduction in energetically expensive activities following the treatment. Lucas et al. (1999) considered the effects of body mass and food availability on both song and non-song vocalisations. They found, in general agreement with theory, that song rates were highest in birds with relatively high dawn mass but that non-song vocalisations were

negatively correlated with body mass and therefore may be a signal used by birds facing energetic stress. Other empirical tests have shown, in European robins (Erithacus rubecula), that both an increase in foraging success and a reduction in the variability of foraging increased the output of display (singing) behaviour (Thomas, 1999a, 1999b). Cuthill et al. (2000) tested the effects of unpredictable food availability on the body mass regulation of European starlings and found that dawn mass, dusk mass and daily mass gain were all increased. With the exception of Dall & Witter (1998), these examples collectively support the modelling predictions that predictable foraging success will lead to decreased body mass and increased display effort, and vice versa.

All of these studies have examined the effects of feeding manipulations on either body mass regulation or courtship activity in isolation but none have looked at the dynamic interaction between body mass regulation and courtship behaviour that is central to the models. By restricting food during distinct periods in an individual's day we explored the short-term consequences for body mass maintenance and display hopping activity with particular reference to the period of restriction. The predictions of Hutchinson et al. (1993) assumed that shortening daylength decreased the energy available to a bird but this did not allow for the possibility of compensatory foraging if feeding conditions were particularly good, and the predictions of McNamara et al. (1987) envisaged the timing of interruptions being variable within and between days. Our experiment did not therefore explicitly test predictions of either model as birds enjoyed *ad libitum* food availability outwith restriction periods and the timing of our restrictions were constant within a treatment. We predicted that during the third of the day when food was unavailable, individuals would seek to limit their mass loss by reducing energetic activities such as display behaviour. However, the extent to which individuals varied the trade-off between body mass maintenance and display activity when faced with food restriction was predicted to vary also with the timing of the restriction. Display behaviour appears to be more important at certain times of day (e.g. dawn) therefore individuals are predicted to guard their performance of that behaviour, at those times of day when it is typically performed most, with the decrease in body mass and increase in starvation risk it would entail.

Methods

Subjects and Housing Conditions

We conducted this experiment using an experimental group of eight male and a stimulus group of eight female zebra finches. The two groups of individuals had no prior experience of one another except during a period of two days prior to the first data being collected when they were introduced into the experimental cages. All birds used in the experiment were sexually mature. Before and during the experiment all individuals were maintained on a 12:12 hour light:dark photoperiod (lights on 0930 to 2130) at a temperature of mean ± S.D. = 19.2 ± 4.5°C. Simulated dawn and dusk was applied during the first and last 30 minutes of the daylight period by means of the sequential switching of a lighting array. The light was provided by eight 40 watt incandescent bulbs in a ceiling-mounted linear array, six ceiling-mounted 58 watt fluorescent tubes with diffusers and two ceiling-mounted 300 watt halogen floodlights. Daylight was completely excluded. During dawn and dusk the light level increased and decreased approximately exponentially (see chapter 3). We provided the birds with *ad libitum* food (Haiths' Foreign Finch Mix, Haiths, Cleethorpes, UK), except during imposed food restrictions (see experimental procedure), and water. Birds were given access to shell-grit and cuttlebone prior to the experiment but not during it.

Experimental Cages

Eight experimental cages (1.5 x 0.3 x 0.4 m high, Figure 1) were employed, each consisting of three equally sized sections: food, courtship and stimulus. In each cage the experimental male had access to the food and courtship sections and the stimulus section was occupied by the same female zebra finch throughout. The food and courtship sections were separated by opaque baffles which permitted the male to move freely from one section to the other but prevented him from simultaneously feeding and being in visual contact with the stimulus section. The courtship and stimulus sections were separated by a transparent Perspex panel (0.15 cm thick). The food section contained a food hopper and water dish together with a microswitch feeding perch that recorded the time and duration of feeding bouts (provided that the bird chose to feed from this perch). The courtship section contained a set of four microswitch perches that

recorded the temporal and spatial patterns of hopping activity. The four perches, made from 5 cm lengths of wooden dowel, were arranged in a square with opposite pairs set 12 cm apart and each perch 14 cm above the cage floor. Male zebra finches hop repeatedly between adjacent perches or perform 'head-tail twists' as part of their courtship display (Morris, 1954; Zann, 1996), and so this apparatus provided a suitable and reliable method of automatically recording display activity (Swaddle & Cuthill, 1994). The perch nearest to the Perspex window that separated the middle and right-hand sections was 4 cm from it. The set of four hop perches was itself mounted on top of an electronic balance (Sartorius PT-610) reading to 0.1 g accuracy. The eight balances were connected to a PC which recorded real-time changes in the body masses of the male birds, and simultaneously corrected for drift from zero in the balances' outputs (see chapter 2).

The stimulus section was equipped with two perches set at the same height as the microswitch perches, one of which was positioned 7 cm and the other 42 cm from the perspex panel. The wall of the right-hand subsection that faced the male's hop perches was white such that stimulus birds were always viewed against a white background. The stimulus section was bedded with composted wood bark. The food and courtship sections however had no substrate and the floor was plain hardboard. This was to prevent excessive debris being deposited by the males on either the perch mechanisms or the balances.

Experimental Procedure

All male birds were subjected in sequence to four food restriction treatments as follows: food available over entire day (treatment 1), food removed for four hours starting at dawn (treatment 2: 0930 to 1330), food removed for middle four hours of day (treatment 3: 1330 to 1730), food removed for last four hours of day (treatment 4: 1730 to 2130). The order of treatments was randomised between cages such that each male experienced each treatment once for a period of seven consecutive days. Treatments followed on from one another without any interim break as any break would have been equivalent to treatment 1. However, to minimise any carry-over effects from one treatment into the next the initial day of every treatment was disregarded in the analyses. The procedure for preventing access to the food involved removal of the food

dish then removal, brushing and replacement of the cage floor. The same procedure was applied for treatment 1 except the food dish was immediately replaced. In the case of treatment 4 the food was removed at the appropriate time then replaced immediately prior to first light the following morning at which time other manipulations were taking place anyway. Since zebra finches do not forage in darkness this was equivalent to replacing the food at onset of darkness but was less disruptive for the birds.

Statistical Analyses

We investigated the effects of food restriction treatments on within-individual variation in body mass, hop rate and the total time spent on the hop perches, using repeated measures ANOVA designs. In addition we investigated the effect of treatment on the diurnal mass trajectory and organisation of hopping behaviour by considering the interaction between treatment and time of day (treatment*time). For the purposes of the analyses the data for each individual were averaged across all six days of each treatment, unless otherwise specified. Body mass data were averaged for 30 minute intervals covering the period of daylight (producing 24 intervals). The number of display hops and total duration of display hops were totalled for the same intervals to produce measures of hop rate and total hop duration per half hour interval, respectively. The ANOVA models therefore specified two factors: treatment (four levels) and time of day (twenty-four levels). The identity of individual males was inherent in the repeated measures design of the model. To determine the effects of food restriction per se on body mass and on hop rate we tested specific contrasts between treatments 1 (control) versus 2, 3 & 4. We also tested pairwise contrasts, for hop rate and body mass, between each food restriction treatment and treatment 1 (control).

To determine if the one day changeover period between adjacent treatments was sufficient or if males continued to modify their display and mass regulation strategies during the course of the treatments we tested a model which included 'day' as a factor with four levels in addition to treatment and time (only days 2 through to 5 were used as the statistical package used would not support this model with more than four days in place).

The repeated measures ANOVA statistics are univariate and where assumptions about sphericity were violated, Huynh-Feldt epilson corrected statistics are cited. Two-tailed tests of probability are used throughout.

Results

There were no effects of treatment on either overall body mass (Fig.2; $F_{3,21} = 1.64$, $P = 0.21$) or display hopping activity (Fig.3; $F_{1.56, 10.96} = 1.06$, $P = 0.36$). We considered specific contrasts, for both body mass and hopping activity, between the three restriction treatments together versus the 'no interruption' (i.e. *ad libitum* food) treatment, and also pairwise contrasts between each of the restriction treatments in turn versus the 'no interruption' treatment (see Table 1). Overall hopping activity and body masses were not significantly different for any of the contrasts. Therefore reducing the availability of food for periods of the day did not affect the mean body mass of the birds or the mean amount of courtship that they displayed each day.

However, there was a highly significant effect of treatment on the temporal pattern of mass over the course of the day (treatment*time; $F_{69,483} = 28.26$, $P < 0.0001$), whereas hopping activity curves were not significantly different between treatments (treatment*time; $F_{4.39,30.74} = 1.57$, $P = 0.20$). We investigated the effect of treatment on the total time spent on the hop perches for each half hour interval through the day. We found that neither the overall amount of time spent on the hop perches (Figure 4; treatment; $F_{1.7, 8.20} = 0.96$, $P = 0.37$) nor the temporal pattern (treatment*time; $F_{10.55,73.82} = 1.65$, $P = 0.10$) varied significantly between treatments. There were marked peaks in the time spent on the perches during the periods when food was being replaced or removed from cages (see Figure 4). As the birds only experienced human intrusion at these times it appears that activity at the food section of the cage displaced birds towards the stimulus section (and therefore the display perches). Similarly, males may have sought the relative security of associating with the female during human intrusions (c.f. Lombardi & Curio, 1985).

The more complex models which included day as a factor revealed that neither overall body mass (treatment*day; $F_{9,63} = 0.94$, $P = 0.50$) nor overall hopping activity (treatment*day; $F_{9,63} = 0.45$, $P = 0.90$) varied over the course of a treatment. These

results suggest that the one day lead into each treatment was sufficient for males to respond to the change in food restriction pattern and that subsequent mass regulation and hopping patterns remained consistent during the treatment.

Discussion

Despite food restrictions during which the daily foraging period was cut by one third, there were no effects on overall body masses of male zebra finches. Nor was there any effect on their hopping activity rates, the temporal organisation of hopping activity, the total time spent per day on the hop perches or the temporal organisation of time spent on the hop perches. Instead the effect of food restriction manifested itself as a highly significant difference in the body mass trajectories of the males. To some extent the significant effect of treatment on the body mass trajectories is not at all surprising. Aside from drinking, which would on average have balanced water loss (Cade et al., 1965), a male could not have gained body mass during a food interruption treatment. The consequence of this, even if his metabolic rate was only sufficient to support life, would have been that his mass would begin to drop. In the present study we saw activity levels that were clearly far above those corresponding to baseline metabolism and consequently we also saw sharp drops in body mass associated with the absence of food in each of the three food restriction treatments. Coupled with these dramatic mass losses were equally dramatic gains once food access was restored: the highest mean mass gain over a half an hour during the control treatment (no interruption to food supply) was 1.51% of initial body mass compared to a peak gain of 3.82% in the 30 minutes following re-introduction of food in a food restriction treatment (i.e. the rate of mass gain more than doubled). These figures indicated that males had a substantial amount of spare gut and/or digestive capacity which was only brought into play under circumstances of intermittent food availability (c.f. Sibly & Calow, 1986; Kersten & Piersma, 1987; Lepczyk et al., 1998; Lepczyk & Karasov, 2000). In starlings, Bautista et al. (1998) have shown that under conditions of energetic stress, the efficiency of food utilisation increased (as indicated by a lower calorific density of faeces). It has also been shown that starlings have the capacity to change gut morphology and improve gut biochemistry in response to changes in nutrient intake (Martinez del Rio et al., 1995).

The fact that many bird species maintain their body masses below what would be physiologically possible is widely held as evidence that carrying fat is costly (for a review see Witter et al., 1993; see also Gosler et al., 1995). It seems reasonable to assume therefore that, since birds maintain their rates of mass gain below those that we have shown are physiologically possible, there must be costs associated with elevated levels of mass compensation. In the absence of any such costs it could be argued that a plausible alternative to the typical diurnal mass trajectory (e.g. Fig. 2, 'no interruption' curve) might be to maintain a relatively low and constant mass throughout the day before sharply increasing in mass during late afternoon to meet a target dusk mass. This strategy would substantially reduce mass-dependent costs through much of the day. However, such a pattern has never been observed in birds either in the wild or in laboratory studies, and it is highly unlikely to be biologically feasible due to the assumption of reliable *ad libitum* food availability and due also to the pathological costs of prolonged periods of extreme rates of mass gain (Sjödin et al., 1990).

Although overall hopping activity levels were not significantly different between treatments, it was particularly surprising that under the dawn interruption treatment the males were second most active, and under the other two interruption treatments the males were at their most active levels, specifically during the period corresponding to the food restriction. However, balancing this was a non-significant pattern of reduced hopping activity in the period immediately after the food-restriction, during which the rate of mass gain rose sharply. Kacelnik & Krebs (1982) argued that a functional explanation for the dawn chorus in passerines may be that foraging is less profitable at twilight owing to low light intensities. Since foraging and displaying are mutually exclusive, time devoted to displaying is time lost from foraging. When the availability of food is low, or as in the present case nil, the disadvantage of lost foraging time is also nil and therefore the net benefit of displaying is substantially increased. Further, body mass will decrease during food restriction, especially if activity is raised, which serves to reduce mass-dependent costs and therefore make the activity less expensive still. While in one sense engaging in display activity during food restriction is therefore cost-free, the heightened metabolic costs will increase the risk of starvation. Without starving a zebra finch to death it is impossible to say exactly how resilient they may be, or at what point the threat of starvation starts to become physiologically relevant. What is clear is that in the present study male zebra finches appeared to overlook the

possibility of starvation in favour of maintaining or (non-significantly) increasing display hopping and were able to compensate rapidly for sustained losses in body mass upon food restoration.

Dall & Witter (1998) examined the responses of zebra finches to unpredictable foraging interruptions in terms of their diurnal mass changes and behavioural routines. That study found that zebra finches did not alter their mass trajectories or their body mass but did respond by reducing energetically expensive activities. The authors suggested that the length of the restriction period relative to the total feeding period (25%) was too short to noticeably increase the perception of food shortage. This was dismissed however, as similar durations evoked mass changes in other studies, and moreover their manipulation was sufficient to significantly affect behaviour in other ways. Another possibility they discussed for the lack of effect on body mass was that the overall duration of the feeding period was too short at just six hours. Again this was dismissed as birds were clearly not gaining mass at their maximal rate over the course of the day which implied that the capacity to modify their mass regulation strategies was available.

Irrespective of the explanation, the result from that study is incongruous with the existing theoretical (Houston & McNamara, 1987, McNamara et al., 1987; Hutchinson et al., 1993) and other empirical (Ekman & Hake, 1990; Bednekoff & Krebs, 1995; Witter et al., 1995; Witter & Swaddle, 1997) literature and also with the present work. There was an important key difference in experimental protocol between that adopted by Dall & Witter (1998) and the present study. Dall & Witter (1998) measured body masses and behaviour for five days before and after a 14 day period when food availability was reduced. Therefore the data being compared were all from individuals experiencing *ad libitum* food supply with the only difference being in their prior experience. The effects of food restriction seen in that study were tentatively attributed to the natural feeding ecology of the zebra finch (Zann & Straw, 1984) and the temporally and spatially unpredictable patterns of grass seed availability (Morton & Davies, 1983). Data from a study of the White Crowned Sparrow (Zonotrichia albicollis) indicated increased locomotory activity during a period of food restriction which was interpreted as an adaptation to changing foraging site more frequently (Ketterson & King, 1977). Dall & Witter (1998) suggested that the activity patterns they saw following the period of restricted food availability were an adaptation to the

unpredictable nature of the species' food resource. Coupled to the possibility of increased locomotion as a foraging adaptation during food shortage (on which they presented no data) is their observed pattern of decreased locomotion accompanying restoration of food supply, as a means of restoring an energy balance.

We tested our data taking into account the effect of day of treatment, using the second to fifth days (inclusive) from each treatment. These results indicated that (for both body mass and hopping rate) the overall effect of food interruption did not differ between days over the duration of the treatment and nor did the effect of treatment with time (i.e. the trajectory). These are important results as they show that the birds' mass regulation strategies and hopping behaviour responded quickly to the changing schedules of food availability and then remained consistent for the whole of each treatment. In other words, the effects were not progressive over the course of the treatment. From this we propose that in male zebra finches, information on current food availability is more important than (even recent) historical information in shaping body mass regulation strategies and hopping activity patterns in the presence of a female (c.f. Cuthill et al., 2000 on starlings). This may explain why Dall & Witter (1998) failed to see an effect of their food restriction treatment on mass regulation strategy since their effect of food restriction was measured during a five day period of *ad libitum* food availability immediately following the 14 day restriction period. Our data would suggest that within approximately one day of returning to *ad libitum* food their finches would have re-modelled their mass regulation strategies and no effects of the previous food restriction period would have remained apparent.

Rogers et al. (1994) discuss three mechanisms by which individuals might reconcile environmental conditions with body mass. These are described as (1) 'predictors' who follow a fixed (e.g. seasonal) pattern of mass change, (2) 'responders' who monitor food availability and adjust mass accordingly and (3) 'predictor-responders' who combine mechanisms (1) & (2). Field studies indicate some evidence of all three of these types (see Dall & Witter, 1998). Given the various mass-dependent costs of carrying food/fat reserves it would appear that if physiology permits, the optimal policy is to make real-time responses to current conditions (i.e. to be a very efficient 'responder'). Consider a case where foraging conditions fluctuate between two states; good or bad. Individuals basing current state on historical information will always be

disadvantaged during the period of change from one foraging condition to the other by being either too light (so incurring a greater risk of starvation) or too heavy (incurring undue costs of feeding and fat storage). As the information used to set current state becomes more recent so the duration of this disadvantage decreases. The ability to respond to current conditions would require extremely plastic body mass regulation such as we saw in the zebra finch with its potential for relatively large daily mass gains and its capacity to achieve high rates of mass gain in the very short term.

In conclusion, our results did not agree with our specific hypotheses formed around the general theoretical predictions of the effect of foraging interruptions on the dynamic trade-off between body mass maintenance and courtship behaviour. Males defended their diel pattern of display activity vigorously and equally between treatments and there was no evidence of food interruptions affecting the trade-off of investment in reserves against display activity. Indeed there was a non-significant tendency for hopping activity to actually be higher or highest specifically during the period of food restriction. This may relate to the net costs of courtship being lower in the absence of a foraging option, irrespective of any variability in the gross benefit of displaying within certain temporal contexts. Males responded quickly to introduction of the treatments and their response remained consistent throughout the duration of the treatment. This implied that current information was more important than historical information in re-modelling diel patterns of courtship and mass regulation, and given the extreme plasticity with which somatic reserves were regulated this seemed to be a very attractive and workable policy.

References

Bautista, L. M., Tinbergen, J., Wiersma, P. & Kacelnik, A. 1998. Optimal foraging and beyond: how starling cope with changes in food availability. American Naturalist, 152, 543-561.

Bednekoff, P. A. & Krebs, J. R. 1995. Great Tit fat reserves: effects of changing and unpredictable feeding day length. Functional Ecology, 9, 457-462.

Cade, T. J. & Tobin, C. A. & Gold, A. 1965. Water economy and metabolism of two estrilidine finches. Physiological Zoology, 38, 9-33.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Cuthill, I. C., Maddocks, S. A., Weall, C. V. & Jones, E. K. M. 2000. Body mass regulation in response to changes in feeding predictability and overnight energy expenditure. Behavioral Ecology, 11, 189-195.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Ekman, J. B. & Hake, M. K. 1990. Monitoring starvation risk: adjustments of body reserves in greenfinches (Carduelis chloris L.) during periods of unpredictable foraging success. Behavioral Ecology, 1, 62-67.

Gosler, A. G., Greenwood, J. J. D. & Perrins, C. 1995. Predation risk and the cost of being fat. Nature, 377, 621-623.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

Kersten, M. & Piersma, T. 1987. High levels of energy expenditure in shorebirds: metabolic adaptations to an energetically expensive way of life. Ardea, 75, 175-187.

Ketterson, E. D. & King, J. R. 1977. Metabolic and behavioral responses to fasting in the white-crowned sparrow (Zonotrichia leucophrys gambelli). Physiological Zoology, 50, 115-129.

Lepczyk, C. A., Caviedes-Vidal, E. & Karasov, W. H. 1998. Digestive responses during food restriction and realimentation in nestling house sparrows (Passer domesticus). Physiological Zoology. 71, 561-573.

Lepczyk, C. A. & Karasov, W. H. 2000. Effect of ephemeral food restriction on growth of house sparrows. The Auk, 117, 164-174.

Lombardi, C. M. & Curio, E. 1985. Social facilitation of mobbing in the Zebra Finch Taeniopygia guttata. Bird Behaviour, 6, 34-40.

Lucas, J. R., Schraeder, A. & Jackson, C. 1999. Carolina chickadees (Aves, Paridae, Poecile carolinensis) vocalization rates: effects of body mass and food availability under aviary conditions. Ethology, 105, 503-520.

Martinez del Rio, C., Brugger, K. E., Rios, J. L., Vergara, M. E. & Witmer, M. 1995. An experimental and comparative study of dietary modulation of intestinal enzymes in European starlings (Sturnus vulgaris). Physiological Zoology, 68, 490-511.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Morton, S. R. & Davies, P. H. 1983. Food of the zebra finch (Poephila guttata) and an examination of granivory in birds of the Australian arid zone. Australian Journal of Ecology, 8, 235-243.

Rogers, C. M., Nolan, V. & Ketterson, E. D. 1994. Winter fattening in the dark-eyed Junco: plasticity and possible interaction with migration trade-offs. Oecologia, 97, 526-532.

Sibly, R. M. & Calow, P. 1986. Physiological Ecology. Blackwell Scientific, Oxford.

Sjödin, B., Westing, Y. H. & Apple, F. S. 1990. Biochemical mechanisms for oxygen free radical formation during exercise. Sports Medicine, 10, 236-251.

Swaddle, J. P. & Cuthill, I. C. 1994. Female zebra finches prefer males with symmetric chest plumage. Proceedings of the Royal Society of London Series B, 258, 267-271.

Thomas, R. J. 1999a. Two tests of a stochastic dynamic programming model of daily singing routines in birds. Animal Behaviour, 57, 277-284.

Thomas, R. J. 1999b. The effect of variability in the food supply on the daily singing routines of European robins: a test of a stochastic dynamic programming model. Animal Behaviour, 57, 365-369.

Witter, M. S. & Cuthill, I. C. 1993. The ecological costs of avian fat storage. Philosophical Transactions of the Royal Society of London B, 340, 73-92.

Witter, M. S. & Swaddle, J. P. & Cuthill, I. C. 1995. Periodic food availability and strategic regulation of body mass in the European Starling, Sturnus vulgaris. Functional Ecology, 9, 568-574.

Witter, M. S. & Swaddle, J. P. 1997. Mass regulation in juvenile Starlings: response to food availability depends on initial body mass. Functional Ecology, 11, 11-15.

Zann, R. & Straw, B. 1984. Feeding Ecology and Breeding of Zebra finches in Farmland in Northern Victoria. Australian Wildlife Research, 11, 533-552.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Table 1. Specific contrasts between treatments for body mass and hopping activity. The control (treatment 1) was compared to each of the other treatments in turn, and to the mean effect of the other three treatments.

		F-value ($_{1,7}$ df)	P-value
Body Mass	1 v 2, 3 & 4	3.83	0.09
	1 v 2	3.70	0.096
	1 v 3	0.60	0.48
	1 v 4	3.61	0.099
Hopping Activity	1 v 2, 3 & 4	0.12	0.75
	1 v 2	0.37	0.56
	1 v 3	1.55	0.25
	1 v 4	1.83	0.22

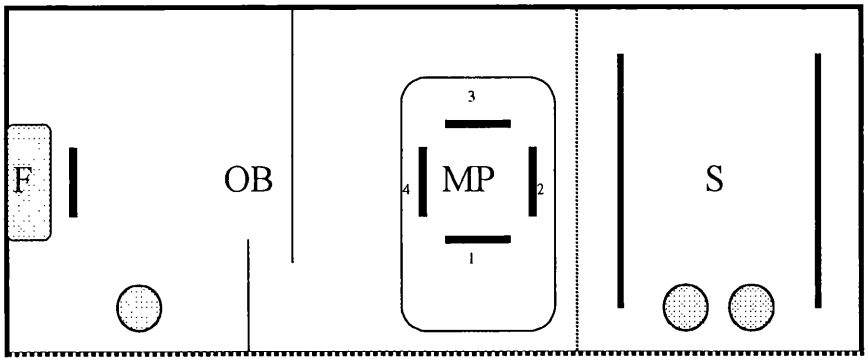


Figure 1. View of the experimental cage layout. Food (left) and Courtship (middle) sections were partially separated by opaque baffles and housed food hopper and electronic balance/microswitch perch mechanism respectively. Stimulus section (right) housed the stimulus bird (a female zebra finch) and was separated from the middle section by a perspex window. Cages were solid on all but the front face, which was wire mesh.

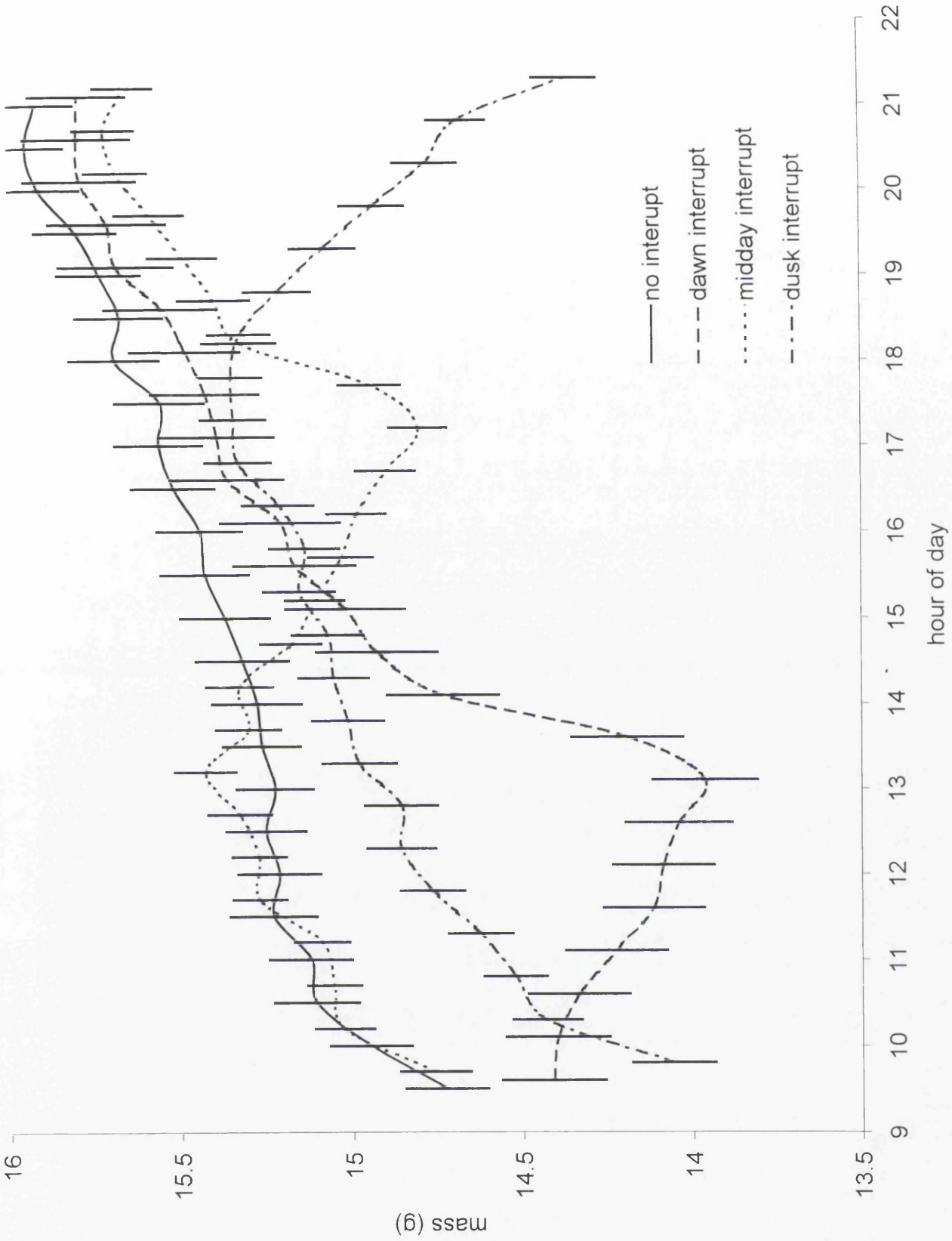


Figure 2. Mean body mass trajectories (±S.E.) of male zebra finches for each of the four treatments. X-axis values are staggered among treatments by 6 minutes for clarity (to prevent error bars from overlapping). Treatment 1, 'no interruption to food'; Treatment 2, 'dawn interruption'; Treatment 3, 'midday interruption'; Treatment 4, 'dusk interruption'.

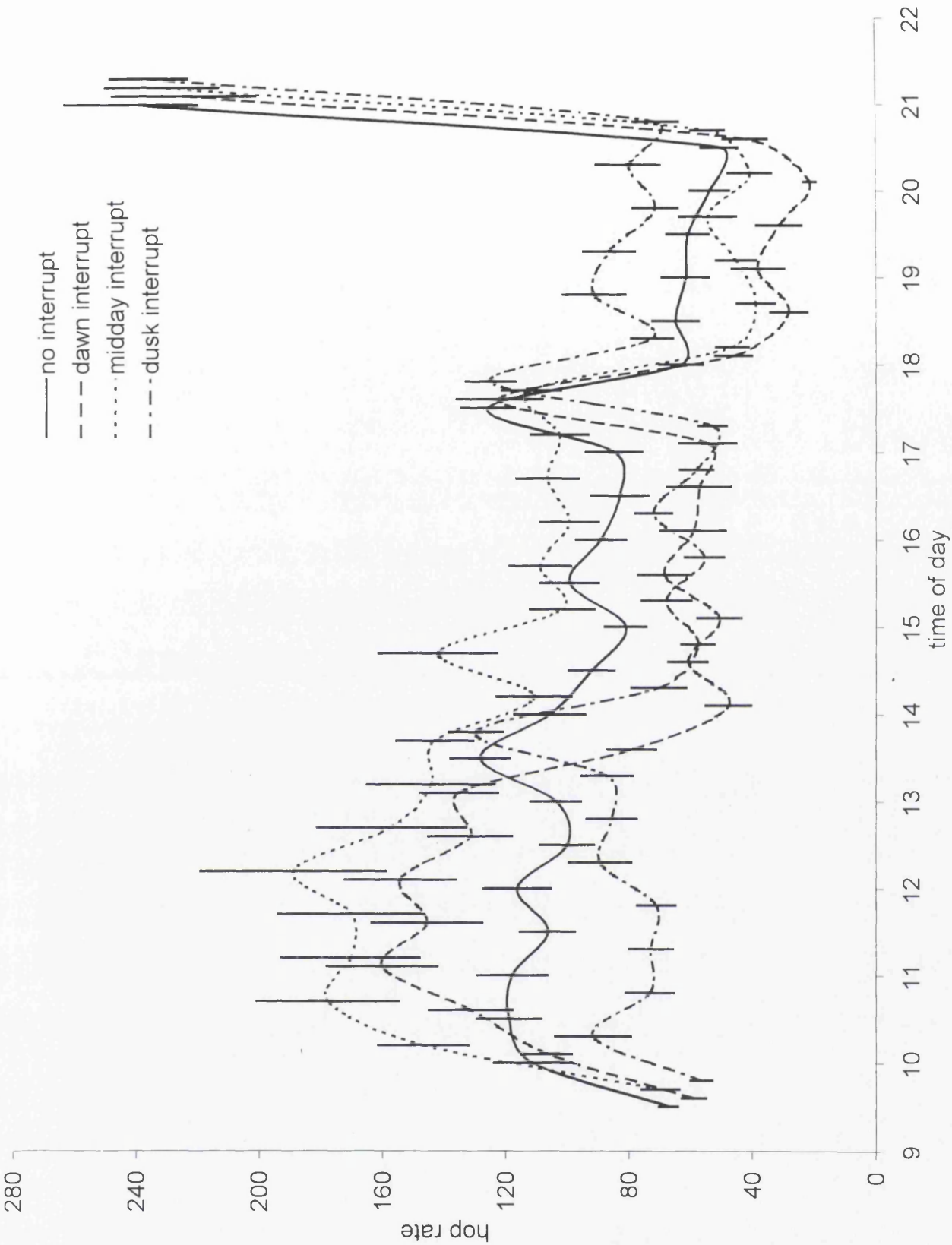


Figure 3. Mean hop rate trajectories (±S.E.) of male zebra finches for each of the four treatments. Data presented as in Fig. 2.

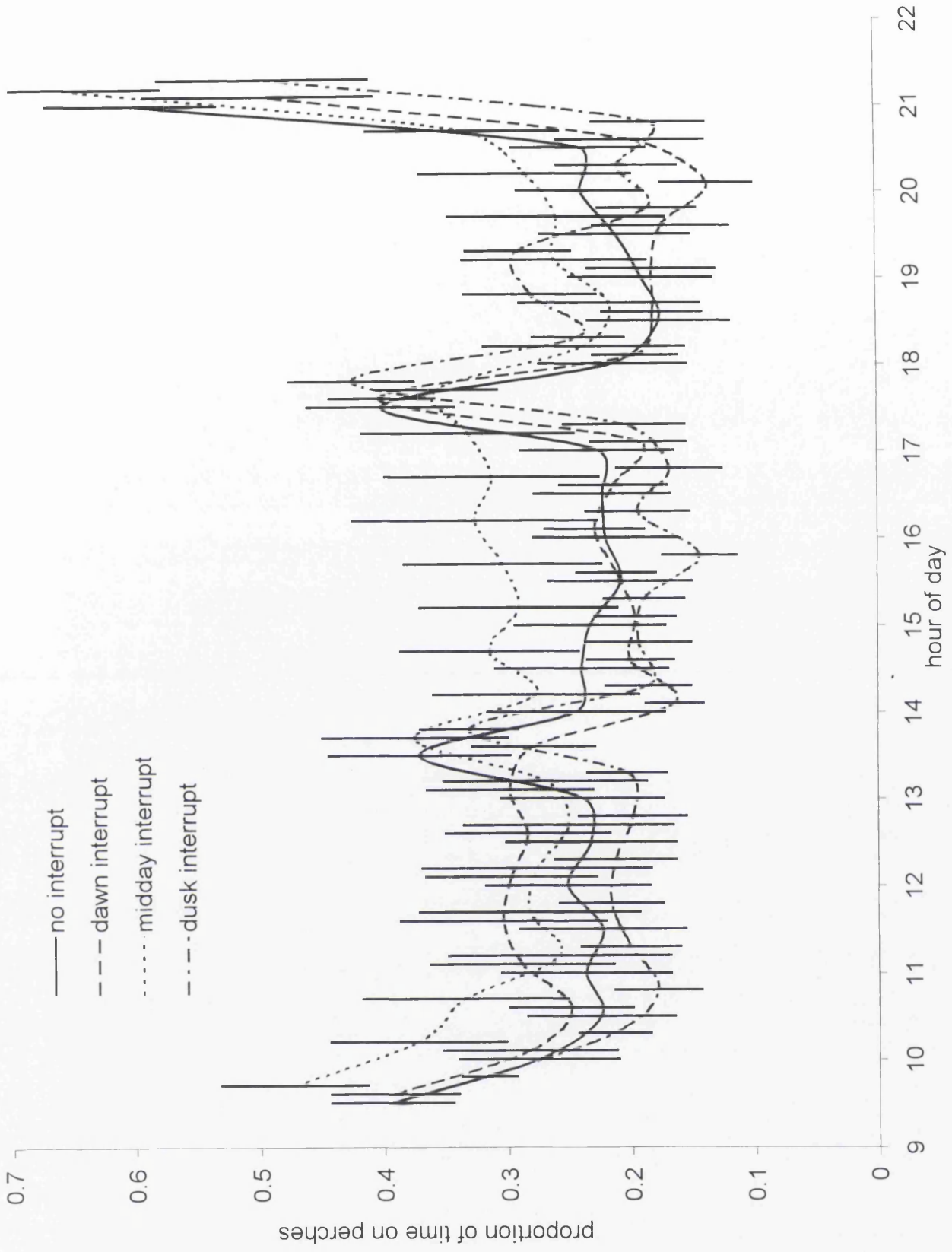


Figure 4. Mean of total time spent on hop display perches (±S.E.) (expressed here as a proportion of each half hour interval) by male zebra finches experiencing each of the four treatments. Data presented as in Fig. 2.

Chapter 8. General Discussion

The predominant theme of this series of experiments was factors affecting the putative dynamic trade-off in the short term allocation of resources to investment in reproduction (courtship activity) versus investment in the soma (body mass regulation), using the male zebra finch as a model. Previously, this trade-off had not been investigated in birds, although theoretical predictions pointed to its probable existence (Houston & McNamara, 1987; McNamara et al., 1987; Hutchinson et al., 1993). Such a trade-off has been shown to exist in mammals, for example Richardson's ground squirrels (Spermophilus richardsonii) (Michener, 1998) and in the red fox (Vulpes vulpes) (Cavallini, 1998), although in both these cases the trade-off was apparent not in the short term, but over the course of an entire breeding season. Given the nature of the trade-off I predicted that it would be influenced by manipulations of the opportunity to forage or conduct courtship (i.e. the presence or absence of a mate, or the state of that mate).

The experiment described in chapter 4 demonstrated clearly that this trade-off existed, an effect never before demonstrated in birds. The experimental design allowed me to specifically test for the effects of the presence/absence of a potential mate (female conspecific), since two further treatments controlled for effects due simply to the presence of another bird. Although males conducted hopping activity in all treatments, performance of that behaviour was significantly higher during the female zebra finch stimulus treatment, and at the same time mean body masses were significantly depressed throughout the day in that treatment, while the diel trajectory of body mass remained unaffected.

This experiment provided an excellent basis for the other work which I performed as it confirmed the existence of a trade-off, and further it indicated that the design of the experimental environment, which was largely unchanged throughout the series of experiments, was both appropriate and sensitive enough to detect this trade-off. As will be evident from chapter 2, a tremendous amount was invested in the design and implementation of the experimental environment, which is one justification for using it repeatedly. However, the principal justification was its suitability for this series of

experiments. The design of the cages created a conflict between display behaviour and foraging behaviour and simultaneously recorded both body mass regulation and display behaviour.

The fact that the food source was close at hand and easily accessible, and that food was available *ad libitum* (for exceptions see chapter 7), could be suggested as possible weaknesses in the design. It could further be argued that such a situation was not necessarily conducive to creating or underlining a trade-off. There are several convincing lines of defence to this argument. The first is that this argument supposes that the only cost of foraging was in time lost from other activities. This was of course only one factor. Because it inevitably leads to increased body mass, foraging incurs numerous mass-dependent costs (e.g. Witter & Cuthill, 1993), including the mass-dependent costs of foraging itself. There are also time-dependent costs of foraging which include exposure to parasites, pathogens or predators (Deerenberg et al., 1998), together with risks associated with leaving a potential mate unattended (e.g. Birkhead et al., 1988; Mace, 1989a; Møller, 1991).

Further, the argument against the sensitivity of the experimental arrangement to evoke or detect a trade-off, inferred that although a male bird may have increased his energy expenditure by displaying to a female, he had a ready supply of food and could therefore compensate by consuming more. This ignored putative short term costs of increasing the rate of food consumption (Sjödin et al., 1990), and it both assumed that individuals will have been able to also digest this food at a higher rate, and ignored the possible costs of changes occurring in gut morphology and/or gut mass (Martinez del Rio et al., 1995). Coupled to this was the possibility of a metabolic ceiling which may have restricted the energy available to an individual over a period of time (Hammond & Diamond, 1992; Suarez, 1996; Rosén et al., 1999). Studies on European starlings (Sturnus vulgaris) (Bautista et al., 1998) and on zebra finches (Dall & Witter, 1998) indicated that those species respectively spent around 90% and 30% to 50% of time during daylight being inactive. Inactivity becomes necessary whenever an animal approaches its limit for aerobic work, but of course simply because an animal chooses to be inactive does not imply that it has reached its aerobic limit. However, in chapter 4 there was the situation where the body masses of birds which were displaying most, were depressed across the whole day. In this situation, the fact that individuals chose

not to maintain their body mass by assigning time to foraging did suggest that there was either a digestive or metabolic limitation acting on them. Assuming that there was an upper limit to the digestive rate or the rate of aerobiosis, there was consequently a finite amount of energy available to an individual which had to be apportioned over the course of a day. If the overall level of activity had risen, as it may have done in those males that increased their rate of display, one means of funding this would have been to reduce metabolic overheads by reducing body mass, and consequently reducing mass-dependent costs also. Lowering body mass throughout the day is consistent with this hypothesis and also indicated that males had an interest in guarding their diel mass trajectories (c.f. Dall & Witter, 1998; Cuthill et al., 2000), an effect also apparent in chapters 5, 6 & 7.

The trade-off between investment in courtship versus investment in body mass regulation was manifested in chapters 5 & 7 as well as in chapter 4. In chapter 5, I investigated time of day changes in the effect of presentation of a female stimulus bird, and by removing the natural association between the time-states of the two sexes I was also able to investigate time of day changes in the females' attractiveness/ responsiveness to the males. The data were analysed such that effects due to the time of presentation could be separated from effects due to the female's time-state. The first analysis considered the differences between a discrete three-hour presentation of the female at dawn versus midday. There was no effect of presentation time on the overall level of hopping activity, although the trajectories were significantly different. In short the males' hopping activity levels were no different overall, but the peak of display was time-shifted to coincide with the time at which the female was present. Although dawn is the time of day when males normally display most (e.g. Kacelnik & Krebs, 1982; Mace, 1987a, 1987b, 1989b; Thomas, 1999a, 1999b), this occurs when the female is present for the whole duration of the day (e.g. chapters 3, 4, 6 & 7). In this case, because the female was presented for one quarter of the daylight period, it appears that the males were maximising their potential gain from that opportunity, irrespective of the time of presentation, which is in keeping with the opportunistic breeding nature of the species (Morris, 1954; Zann, 1996).

Although there was no effect of presentation time on overall hopping activity levels (chapter 5, figure 6), there was a significant effect on males' mean body masses, with

lower masses during the 'dawn presentation' treatment. At first glance this may seem incongruous; however, there were clear differences in the state of the males during their respective display bouts at dawn and at midday. At dawn the male had just emerged from a 12 hour fast and his relative body mass, fat/food reserves and gut contents would all have been at their lowest point in the diurnal cycle, and certainly lower than at midday. Given these differences in state, the relative costs of the display bouts (at dawn versus midday) would also have differed, with the risk of starvation for the male displaying at dawn being higher. One means of offsetting part of this increased risk would have been to reduce the display output at dawn, but this would have restricted the males' reproductive opportunities (e.g. Mace, 1987b; Cuthill & MacDonald, 1990). An alternative would have been to reduce mass-dependent costs (see Witter & Cuthill, 1993) in general and specifically those of the display activity, by reducing overall body mass, and such an effect was apparent (chapter 5, figure 3).

Also in chapter 5, I considered the effect of time of day changes in the females' attractiveness/responsiveness on male behaviour, independently of the time at which females were presented to the male. Males displayed significantly more to females who were in the dawn-state, but only when those females were presented at dawn, making this a context-dependent effect. Female fertility has been shown to peak around dawn (Mace, 1987b) which in turn favours courtship and mate guarding behaviour at this time (Mace, 1989a). This experiment indicated that the attractiveness/ responsiveness of females was higher at dawn than midday, and the context-dependence of the effect implied that there was also a time of day variation in the behavioural interaction between the male and female. Similar to the case for the time of presentation, there was a tendency (although in this case a non-significant one) for body mass to be lower under the treatment during which most display behaviour occurred (i.e. the 'dawn-state' treatment), which again may have been a means of financing increased energy expenditure by reducing mass-dependent costs. No corresponding differences were discernible between male displays during midday, to either female time-state treatment.

Similar effects are apparent in chapters 4 & 5, insofar as an increase in the investment in reproduction was traded off principally against the overall mean body mass of the male, with the shape of the mass regulation trajectories tending to be 'defended' (i.e. kept constant). Chapter 7, which was concerned with the effects of food interruptions on

male courtship effort and mass regulation, showed that male zebra finches had the ability to more than double their usual rate of mass gain, at least in the short term. These dramatic mass gain rates occurred upon restoration of food following a period when food was absent, which lasted for a third of the feeding day. What was interesting here was that when faced with this energetic stress, they did not reduce but in fact tended to increase (non-significantly) their level of display hopping activity, specifically during the period when food was absent. A possible reason for this may be that when food availability was nil, the cost of courtship, measured in the currency of lost foraging opportunities, was also nil (c.f. Kacelnik & Krebs, 1982).

What was also interesting, and related to the results from chapters 4 & 5, was that despite the significant disruption that the food restriction treatments caused to the body mass trajectories, males appeared to prioritise the restoration of their mean mass and mass trajectory. During the period when food was absent a male had no means, aside from drinking, of maintaining his mass trajectory, and as he continued to metabolise food his body mass inevitably decreased. However, upon restoration of the food supply, males adopted a 'catch up' strategy of a high rate of mass gain for a brief period of two to three hours before resorting to the more usual (compared, for example, to the control treatment) rate of mass gain.

This recovery response of energetically stressed birds and 'defence' of the diel mass trajectory, is likely to be an adaptive mechanism and may have been mediated by endocrine secretions of corticosterone (e.g. Harvey et al., 1984; Schwabl et al., 1985). Astheimer et al. (1992) performed a series of experiments in which they examined the effects of plasma corticosterone on feeding behaviour following a period of fasting. In one experiment white-crowned sparrows (Zonotrichia leucophrys) and song sparrows (Z. melodia) were implanted with sources of corticosterone and their responses to a 24 hour fast were measured. Upon food restoration, implanted birds fed for longer and at a higher intensity than control birds. In a second study, dark-eyed juncos (Junco hyemalis) had higher endogenous levels of corticosterone following a 24 hour fast compared to a control group which had been allowed access to food for 1 hour upon completion of their fast. Astheimer et al. (1992) concluded that corticosterone secretions were an important endocrine mechanism for initiating/encouraging foraging during periods of food deprivation. The fact that only small increases in the level of circulating

corticosterone were required to evoke intense foraging would have facilitated a rapid return to homeostasis once food was located.

An alternative to the two-step strategy that the males adopted of guarding mass trajectory following food restriction, would have been for them to have a linear (but higher than normal) rate of mass gain from the point of food restoration until the end of the feeding day. There are a number of possible reasons why this may not have been the mode of choice and the observed pattern of mass regulation arose instead. The length of time for which an individual could sustain an above-average level of mass gain may have been limited by factors such as pathological costs of high energy turnover (Harvey et al., 1984; Sjödin et al., 1990), a digestive bottleneck (Sibly & Calow, 1986; Kersten & Piersma, 1987; Lepczyk et al., 1998; Lepczyk & Karasov, 2000) or limitations of the gut biochemistry (Martinez del Rio et al., 1995; Suarez, 1996; Bautista et al., 1998). Also, individuals may have perceived the food supply to be stochastic, even though the timing and duration of restrictions were not variable within a treatment, and they may therefore have prioritised rapid short-term gain as an insurance against future restrictions (McNamara et al., 1987). Theoretical models of avian mass regulation and courtship strategies (Houston & McNamara, 1987; McNamara et al., 1987; Hutchinson et al., 1993) assume the existence of a threshold point of body reserves above which an individual should display and below which it should forage, and predict that reserves should oscillate around this threshold. Theory would predict that an individual should be intolerant of any deviation from this threshold. Therefore, the pattern of high mass gain rate followed by a return to a more typical rate, could be viewed as a mechanism to return birds to this threshold level as rapidly as possible, a mechanism which is wholly compatible with an endocrine mechanism mediated by circulating corticosterone.

The experiment described in chapter 6 tested the effect of arbitrary symmetric and asymmetric traits (applied in the form of coloured leg-bands) on the dynamic trade-off between body mass regulation and display hopping activity in male zebra finches. A body of literature has demonstrated in zebra finches, (Swaddle & Cuthill, 1994b; Bennett et al., 1996; Swaddle, 1996; Waas & Wordsworth, 1999; but see Jennions, 1998 for an exception) and in the bluethroat, (Luscinia s. svecica) (Fiske & Amundsen, 1997) that coloured leg-bands applied in symmetric patterns make males more attractive to females. Hansen et al. (1999) demonstrated also that male bluethroats preferred

symmetrically banded females, although the methodology of both bluethroat studies, specifically the way in which colour bands were applied, differed from the methodology adopted here and elsewhere (see Rohde et al., 1997 & Swaddle, 1997 for a discussion). While the extent of any preferences of male zebra finches for arbitrary symmetrical traits in females has never been tested, male mate choice has been demonstrated in zebra finches (Burley, 1981; Burley et al., 1982; Ten Cate & Mug, 1984; Monaghan et al., 1996).

We found no effects of arbitrary symmetry of stimulus females on the diel patterns of courtship and body mass regulation in male zebra finches, and in fact the trajectories for both mass regulation (chapter 6, figure 2) and display hopping activity (chapter 6, figure 3) were highly consistent between treatments. From this, two possible conclusions could be reached. The first is that male zebra finches are not attentive to arbitrary symmetrical traits (or if there are any preferences then they are very weak). The second is that in the present experiment, because females were presented singly, males displayed equally to all females irrespective of their leg-band arrangement, and therefore no firm conclusions about the effects of symmetry can be drawn. However, a separate analysis looked at the effect of individual females (irrespective of leg-band arrangements) on male mass regulation and hopping activity and found a significant effect of female identity on male hopping activity. This indicated that males did not display to all females equally, despite their having been presented singly. The results from chapter 4 confirmed that the trade-off between body mass and investment in reproduction existed, and that the experimental design was able to detect it. Chapter 5 then confirmed that natural variation in female attractiveness/responsiveness affected this trade-off, and again that the experimental design could evoke and detect these differences. Given this evidence it seems most likely that male zebra finches were not attentive to arbitrary symmetry or that any preferences were very weak, and were over-ridden by preferences for natural secondary sexual traits.

Chapter 3 describes an experiment designed to test whether a novel lighting apparatus would have any influence on body mass or hopping activity trajectories. There were no significant effects of lighting regime on either the overall levels or the trajectories of body mass and display hopping activity. However, a pairwise contrast between the rectangular treatment and the ensuing (second) sigmoidal treatment indicated a

significant decrease in body mass. This effect may have been due to a systematic factor such as change in ambient temperature (c.f. Lehikoinen, 1987; Meijer et al., 1996), although such an explanation may have expected a change in mass between the first two treatments, which was not apparent (although there was a slight non-significant increase). A record of temperatures would have allowed this issue to be clarified but unfortunately this was the only study of those described in this thesis where detailed temperature measurements were not recorded. Alternatively, the rectangular lighting regime, which presented no extrinsic cues relating to the beginning or end of the feeding day, may have created an impression of stochasticity as birds had no indication of when their foraging opportunities would end. Certainly, theory would predict that a change from a stochastic to a predictable food supply would cause individuals' mean body masses to decrease (McNamara et al., 1987; Hutchinson et al., 1993), as occurred in the changeover from a rectangular to a sigmoidal regime. This explanation is somewhat incompatible with the fact that mean body masses did not significantly increase when feeding day length switched from being predictable (first sigmoidal treatment) to perceptibly stochastic (rectangular treatment). However, there is a small non-significant increase in body mass during this transition which lends some support to this idea.

The preferred explanation relates to the fact that following the rectangular treatment, the males had five days to re-acclimatise to the new lighting conditions of the second sigmoidal treatment, whereas before the first sigmoidal treatment, the males had the five experimental acclimation days in addition to the 28 days prior to the experiment during which the lighting conditions had also been sigmoidal. Passerines do not feed in darkness or very dim light (Kacelnik, 1979). Therefore, the periods nearest to dawn and dusk during the sigmoidal treatments, when light intensities were lowest, may have shortened the perceived length of the feeding day. However, feeding would have been possible for the full fourteen hours of the rectangular treatment. It was recently demonstrated that zebra finches decreased their body masses in response to shortening daylength, and vice versa (Meijer et al., 1996). The observed decrease in body mass between treatments 2 and 3 may therefore have been a reaction to a perceived shortening of the feeding day length following the transition from the rectangular to the sigmoidal lighting regime. During the 28 acclimation days prior to the experiment, males may have adapted to the prevailing sigmoidal lighting conditions and learned to utilise the periods of lower light around dawn and dusk more effectively. The effect of

this would have been that following the first transition, from sigmoidal to rectangular lighting, the change in perceived feeding day would have been relatively less, and therefore the expected change in mass (in this case an increase) would also have been less.

The mass trajectory during the rectangular regime, although not significantly different from those of the sigmoidal regimes, was slightly higher overall and showed certain peculiarities. In particular, the initial rate of mass gain at dawn tended to be higher (chapter 3, figure 2). Kacelnik (1979) suggested that foraging at dawn under natural conditions may be less profitable due to low light levels. The higher rate of mass gain at dawn during the rectangular treatment may therefore be due to the instantaneous full light intensity promoting an earlier and/or increased foraging effort. The pattern of body mass regulation during the rectangular treatment also suggested a guarded approach. During the rectangular treatment, following a midday plateau, males increased their mass nearly linearly up until just before the onset of darkness. This is in contrast to the sigmoidal treatments where males delayed a final push towards their target dusk mass until around three hours from darkness. This change in the rate of mass gain under the sigmoidal treatments was too early to be directly triggered by any lighting cues (as 'dusk' only began to descend 90 minutes from darkness). However, the fact that the birds postponed the increase until later, compared to during the rectangular treatment, was perhaps due to the fact that there was a warning of imminent darkness that would have permitted adjustments to body mass if required. This experiment also indicated that during the rectangular regime, individuals were able to anticipate the end of the feeding day in the absence of cues linked to decreasing light levels. Patterns of display hopping were very similar to those during the sigmoidal regimes. The sudden upturn in display rate, at about an hour before darkness, indicated that males had an intrinsic means of measuring daylength, which was probably a circadian clock (e.g. Hau & Gwinner, 1992, 1994, 1996, 1997).

The simulated dawn and dusk provided by the lighting apparatus described in chapter 3 was employed in all other experiments except that described in chapter 5, where it was not compatible with the experimental design. The system was relatively inexpensive to build, and provided lighting conditions at dawn and dusk very like those of nature and

very different to those provided by the rectangular lighting regimes that are in common usage in indoor aviaries and animal houses.

Future Work

The work described in chapter 5 makes a significant contribution to the understanding of time of day effects on male-female courtship encounters by demonstrating that males displayed more to females whose internal time-state was set to dawn. Furthermore, this effect was context dependent insofar as the preference for dawn-state females was only apparent during the males' dawn. This experiment was, however, unable to differentiate between time of day changes in the attractiveness versus the responsiveness of the stimulus female. Indeed, the two effects are very closely related as courtship in zebra finches is strongly interactive therefore being responsive could be construed as also being attractive (Morris, 1954; Immelmann, 1959 & 1962 as cited by Zann, 1996). A worthwhile extrapolation of this study would be to attempt to separate the effects of time of day changes in female attractiveness and responsiveness. This could be studied by removing the interaction between the male and female, such that the male could view the female but she could not view or respond to him. To achieve this it would be necessary to replace the wire screen between the courtship and stimulus sections with one-way glass, and to remove or limit auditory contact between the experimental and stimulus birds. One-way glass windows have been used in previous studies to eliminate interaction between experimental and stimulus birds (e.g. Sullivan, 1994; Swaddle & Cuthill, 1994a, 1994b; Waas & Wordsworth, 1999). Waas & Wordsworth (1999) highlighted a potential problem with the one-way glass used by Swaddle & Cuthill (1994a, 1994b), which was that the experimental bird, although unable to see the stimulus bird, was confronted with its own mirror image reflected by the one-way glass. This could have a significant influence on the experimental bird's behaviour (e.g. Møller & Höglund, 1991) but can be eliminated by making certain design changes (see Waas & Wordsworth, 1999). Using such an experimental design would remove effects due to female responsiveness and enable any changes in male display effort or mass regulation to be reliably attributed to female attractiveness as opposed to her responsiveness.

Chapter 6 indicated that the display and mass dynamics of male zebra finches were not affected by presenting males with females that differed in an arbitrary symmetrical trait. From this I drew the tentative conclusion that arbitrary symmetry of female zebra finches did not affect their attractiveness to males, in contrast to the same situation in bluethroats (Hansen et al., 1999), or in zebra finches when the sex roles were reversed. To provide a conclusive answer to the question of whether arbitrary symmetrical traits of females influences male choice in zebra finches, it would be necessary to adopt a different experimental design. Previous studies that have investigated the effects of symmetry on mate choice have presented stimuli in two or four-way choice chamber tests (e.g. Swaddle & Cuthill, 1994a, 1994b; Bennett et al., 1996; Swaddle, 1996; Fiske & Amundsen, 1997; Jennions, 1998; Waas & Wordsworth, 1999; Hansen et al., 1999). Such approaches were deemed necessary because they gave the choosing sex the opportunity to reject unfavourable stimuli without completely refraining from courtship activity. In the present study however, a choice chamber test would have ruled out any investigation of mass dynamics in general and specifically whether investment in body mass was traded-off against display effort. Therefore, the experimental design described in chapter 6 was appropriate to test the specific hypothesis that the putative attractiveness of females would influence the dynamic trade-off between investment in courtship versus body mass. However, it would be beneficial if an experiment to test the effects of female arbitrary symmetry on male choice in zebra finches was performed using a choice chamber design in order to provide a definitive answer to the question of whether arbitrary symmetrical traits of female zebra finches influence their attractiveness to males.

Closing Summary

In summary, there has been shown to be a trade-off between investment in reproductive behaviour versus somatic investment in male zebra finches. Males were sensitive to natural time of day variation in female attractiveness/responsiveness and this along with time of day variation in male-state was shown to influence the dynamics of the trade-off. Manipulating the appearance of stimulus females had no effect on the trade-off which led to the conclusion that males are unlikely to be attentive to arbitrary symmetrical traits of females, in contrast to the case when the sex roles are reversed. When faced with food restrictions, males chose to guard or even increase their courtship

behaviour at the expense of behaviours that would have limited mass loss. Male zebra finches demonstrated that their ability to regulate body mass was very plastic in the short term, which may have implications for the way in which they manage their somatic reserves relative to environmental conditions.

References

Astheimer, L. B., Buttemer, W. A. & Wingfield, J. C. 1992. Interactions of corticosterone with feeding, activity and metabolism in passerine birds. Ornis Scandinavica, 23, 355-365.

Bautista, L. M., Tinbergen, J., Wiersma, P. & Kacelnik, A. 1998. Optimal foraging and beyond: how starling cope with changes in food availability. American Naturalist, 152, 543-561.

Bennett, A. T. D., Cuthill, I. C., Partridge, J. C. & Maier, E. J. 1996. Ultraviolet vision and mate choice in zebra finches. Nature, 380, 433-434.

Birkhead, T. R., Clarkson, K. & Zann, R. 1988. Extra-pair Courtship, Copulation and Mate Guarding in Wild Zebra Finches Taeniopygia guttata. Animal Behaviour, 36, 1853-1855.

Burley, N. 1981. Sex Ratio Manipulation and Selection for Attractiveness. Science, 211, 721-722.

Burley, N., Krantzberg, G. & Radman, P. 1982. Influence of colour-banding on the conspecific preferences of zebra finches. Animal Behaviour, 30, 444-455.

Cavallini, P. 1998. Differential investment in mating by red foxes. Journal of Mammology, 79, 215-221.

Cuthill, I. C. & MacDonald, W. A. 1990. Experimental manipulation of the dawn and dusk chorus in the blackbird Turdus merula. Behavioural Ecology and Sociobiology, 26, 209-216.

Cuthill, I. C., Maddocks, S. A., Weall, C. V. & Jones, E. K. M. 2000. Body mass regulation in response to changes in feeding predictability and overnight energy expenditure. Behavioral Ecology, 11, 189-195.

Dall, S. R. X. & Witter, M. S. 1998. Feeding interruptions, diurnal mass change and daily routines of behaviour in the zebra finch. Animal Behaviour, 55, 715-725.

Deerenberg, C., Overkamp, G. J. F., Visser, G. H. & Dann, S. 1998. Compensation in resting metabolism for experimentally increased activity. Comparative Physiology B, 168, 507-512.

Fiske, P. & Amundsen, T. 1997. Female bluethroats prefer males with symmetric colour bands. Animal Behaviour, 54, 81-87.

Hansen, L. T. T., Amundsen, T. & Forsgren, E. 1999. Symmetry: attractive not only to females. Proceedings of the Royal Society of London Series B, 266, 1235-1240.

Harvey, S., Phillips, J. G., Rees, A. & Hall, T. R. 1984. Stress and adrenal function. Journal of Experimental Zoology, 232, 633-645.

Hau, M. & Gwinner, E. 1992. Circadian entrainment by feeding cycles in house sparrows, Passer domesticus. Journal of Comparative A, 170, 403-409.

Hau, M. & Gwinner, E. 1994. Melatonin facilitates synchronization of sparrow circadian rhythms to light. Journal of Comparative Physiology A, 175, 343-347.

Hau, M. & Gwinner, E. 1996. Food as a circadian zeitgeber for house sparrows: The effect of different food access durations. Journal of Biological Rhythms, 11, 196-207.

Hau, M. & Gwinner, E. 1997. Adjustment of house sparrow circadian rhythms to a simultaneously applied light and food zeitgeber, Physiology & Behavior, 62, 973-981.

Houston, A. I. & McNamara, J. M. 1987. Singing to attract a mate: a stochastic dynamic game. Journal of Theoretical Biology, 129, 57-68.

Hutchinson, J. M. C., McNamara, J. M. & Cuthill, I. C. 1993. Song, sexual selection, starvation and strategic handicaps. Animal Behaviour, 45, 1153-1177.

Immelmann, K. 1959. Exerimentelle untersuchungen über die biologische bedeutung artsspezifischer Merkmale beim Zebrafinken (Taeniopygia guttata). Zoologische Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 86, 437-592.

Immelmann, K. 1962. Beiträge zu einer vergleichenden Biologie australische Prachtfinken (Spermestidae). Zoologische Jahrbücher Abteilung für Systematik Okologie und Geographie der Tiere, 90, 1-196.

Jennions, M. D. 1998. The effect of leg band symmetry on female-male association in zebra finches. Animal Behaviour. 55, 61-67.

Kacelnik, A. & Krebs, J. R. 1982. The dawn chorus in the great tit (Parus major): proximate and ultimate causes. Behaviour, 83, 287-309.

Kacelnik, A. 1979. The foraging efficiency of great tits (Parus major) in relation to light intensity. Animal Behaviour, 27, 237-241.

Kersten, M. & Piersma, T. 1987. High levels of energy expenditure in shorebirds: metabolic adaptations to an energetically expensive way of life. Ardea, 75, 175-187.

Lehikoinen, E. 1987. Seasonality of the daily weight cycle in wintering passerines and its consequences. Ornis Scandinavica, 18, 216-226.

Lepczyk, C. A. & Karasov, W. H. 2000. Effect of ephemeral food restriction on growth of house sparrows. The Auk, 117, 164-174.

Lepczyk, C. A., Caviedes-Vidal, E. & Karasov, W. H. 1998. Digestive responses during food restriction and realimentation in nestling house sparrows (Passer domesticus). Physiological Zoology. 71, 561-573.

Mace, R. 1987b. Why do birds sing at dawn? Ardea, 75, 123-132.

Mace, R. 1989a. Great tits choose between food and proximity to a mate: the effect of time of day. Behavioural Ecology and Sociobiology, 24, 285-290.

Mace, R. 1989b. The relationship between daily routines of singing and foraging: an experiment on captive great tits (Parus major). Ibis, 131, 415-420.

Mace, R.1987a. The dawn chorus in the great tit Parus major is directly related to female fertility. Nature, 330, 745-746.

Martinez del Rio, C., Brugger, K. E., Rios, J. L., Vergara, M. E. & Witmer, M. 1995. An experimental and comparative study of dietary modulation of intestinal enzymes in European starlings (Sturnus vulgaris). Physiological Zoology, 68, 490-511.

McNamara, J. M., Mace, R. H. & Houston, A. I. 1987. Optimal daily routines of singing and foraging in a bird singing to attract a mate. Behavioural Ecology and Sociobiology, 20, 399-405.

Meijer, T., Rozman, J., Schulte, M. & Stach-Dreesmann, C. 1996. New findings in body mass regulation in zebra finches (Taeniopygia guttata) in response to photoperiod and temperature. Journal of Zoology, 240, 717-734.

Michener, G.R. 1998. Sexual differences in reproductive effort of Richardson's ground squirrels. Journal of Mammalogy, 79, 1-19.

Møller, A. P. & Höglund, J. 1991. Patterns of fluctuating symmetry in avian feather ornaments: implications for models of sexual selection. Proceedings of the Royal Society of London Series B, 245, 1-5.

Monaghan, P., Metcalfe, N. B. & Houston, D. C. 1996. Male finches selectively pair with fecund females. Proceedings of the Royal Society of London Series B, 263, 1183-1186.

Morris, D. 1954. The reproductive behaviour of the Zebra Finch (Poephila guttata) with special reference to pseudofemale behaviour and displacement activities. Behaviour, 7, 1-31.

Rohde, P. A., Amundsen, T. & Fiske, P. 1997. Fluctuating asymmetry, mate choice and experimental designs. Animal Behaviour, 54, 1030-1033.

Rosén, M., Hedenström, A., Badami, A., Spina, F. & Åkesson, S. 1999. Hunting flight behaviour of the Eleonora's Falcon Falco eleonorae. Journal of Avian Biology, 30, 342-350.

Schwabl, H., Wingfield, J. C. & Farner, D. S. 1985. Influence of winter on endocrine state and behaviour in European blackbirds (Turdus merula). Zeitschrift für Tierpsychologie, 68, 244-252.

Sibly, R. M. & Calow, P. 1986. Physiological Ecology. Blackwell Scientific: Oxford.

Sjödin, B., Westing, Y. H. & Apple, F. S. 1990. Biochemical mechanisms for oxygen free radical formation during exercise. Sports Medicine, 10, 236-251.

Suarez, R. K. 1996. Upper limits to mass-specific metabolic rates. Annual Review of Physiology. 58, 583-605.

Sullivan, M. S. 1994. Discrimination among males by female zebra finches based on past as well as current phenotype. Ethology, 96, 97-104.

Swaddle, J. P. & Cuthill, I. C. 1994a. Preference for symmetric males by female zebra finches. Nature, 367, 165-166.

Swaddle, J. P. & Cuthill, I. C. 1994b. Female zebra finches prefer males with symmetric chest plumage. Proceedings of the Royal Society of London Series B, 258, 267-271.

Swaddle, J. P. 1996. Reproductive success and symmetry in zebra finches. Animal Behaviour, 51, 203-210.

Swaddle, J. P. 1997. Experimental design and the signalling properties of fluctuating asymmetry. Animal Behaviour, 54, 1034-1037.

Ten Cate, C. & Mug, G. 1984. The development of mate choice in Zebra Finches. Behaviour, 90, 125-150.

Thomas, R. J. 1999a. Two tests of a stochastic dynamic programming model of daily singing routines in birds. Animal Behaviour, 57, 277-284.

Thomas, R. J. 1999b. The effect of variability in the food supply on the daily singing routines of European robins: a test of a stochastic dynamic programming model. Animal Behaviour, 57, 365-369.

Waas, J. R. & Wordsworth, A. F. 1999. Female zebra finches prefer symmetrically banded males, but only during interactive mate choice test. Animal Behaviour, 57, 1113-1119.

Witter, M. S. & Cuthill, I. C. 1993. The ecological costs of avian fat storage. Philosophical Transactions of the Royal Society of London B, 340, 73-92.

Zann, R. A. 1996. The Zebra Finch: Synthesis of Field and Laboratory Studies. New York: Academic Press.

Appendix 1

Full listing of programming code (in Delphi/ Object Pascal) for the **DataLog** program described in Chapter 2.

```
 1: unit Loggerunit1;
 2:
 3: interface
 4:
 5: uses
 6:   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 7:   VSSComm32, StdCtrls, ExtCtrls, ComCtrls, Menus, displayInd;
 8:
 9: type
10:   TForm1 = class(TForm)
11:     Panel1: TPanel;
12:     OpenPorts: TButton;
13:     Label6: TLabel;
14:     Panel2: TPanel;
15:     Edit1: TEdit;
16:     Edit2: TEdit;
17:     Edit3: TEdit;
18:     Edit4: TEdit;
19:     Edit5: TEdit;
20:     Edit6: TEdit;
21:     Edit7: TEdit;
22:     Edit8: TEdit;
23:     lblch1: TLabel;
24:     lblch3: TLabel;
25:     lblch5: TLabel;
26:     lblch7: TLabel;
27:     comm1: TVSSComm32;
28:     comm2: TVSSComm32;
29:     comm3: TVSSComm32;
30:     comm4: TVSSComm32;
31:     comm5: TVSSComm32;
32:     comm6: TVSSComm32;
33:     comm7: TVSSComm32;
34:     comm8: TVSSComm32;
35:     Timer1: TTimer;
36:     MainMenu1: TMainMenu;
37:     File1: TMenuItem;
38:     CreateClearFiles1: TMenuItem;
39:     Label5: TLabel;
40:     Exit1: TMenuItem;
41:     Edit20: TEdit;
42:     UpDown2: TUpDown;
43:     BtnSuspend: TButton;
44:     Label9: TLabel;
45:     Label7: TLabel;
46:     About1: TMenuItem;
47:     Panel4: TPanel;
48:     Label11: TLabel;
49:     Label12: TLabel;
50:     Label13: TLabel;
51:     Label14: TLabel;
52:     Label15: TLabel;
53:     Label16: TLabel;
54:     Label17: TLabel;
55:     Label18: TLabel;
56:     Label19: TLabel;
57:     Label20: TLabel;
58:     Timer2: TTimer;
59:     Label21: TLabel;
60:     Image1: TImage;
61:     Mins: TMenuItem;
62:     Secs: TMenuItem;
63:     Exit2: TMenuItem;
64:     Label22: TLabel;
65:     Label23: TLabel;
66:     MenuHint: TMenuItem;
67:     Shape1: TShape;
68:     ShowPerchIndicators1: TMenuItem;
69:     Panel6: TPanel;
70:     Label28: TLabel;
71:     Label29: TLabel;
72:     Label30: TLabel;
```

```
    Label31: TLabel;
    Label32: TLabel;
    Label33: TLabel;
    Label34: TLabel;
    Label35: TLabel;
    Label36: TLabel;
    BeepClick: TMenuItem;
    Label1: TLabel;
    Label2: TLabel;
    Label46: TLabel;
    Label53: TLabel;
    Label54: TLabel;
    Label55: TLabel;
    Label56: TLabel;
    Label57: TLabel;
    edit1tare: TEdit;
    edit3tare: TEdit;
    edit4tare: TEdit;
    edit2tare: TEdit;
    edit7tare: TEdit;
    edit5tare: TEdit;
    edit6tare: TEdit;
    edit8tare: TEdit;
    edit8log: TEdit;
    edit6log: TEdit;
    edit4log: TEdit;
    edit3log: TEdit;
    edit5log: TEdit;
    edit2log: TEdit;
    edit7log: TEdit;
    edit1log: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    Label8: TLabel;
    Label10: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    procedure OpenPortsClick(Sender: TObject);
    procedure comm1ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure ExitClick(Sender: TObject);
    procedure Comm2ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm3ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm4ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm5ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm6ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm7ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure comm8ReceiveData(Buffer: Pointer; BufferLength: Word);
    procedure Timer1Timer(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure CreateClearFiles1Click(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Timer2Timer(Sender: TObject);
    procedure BtnSuspendClick(Sender: TObject);
    procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);
    procedure About1Click(Sender: TObject);
    procedure MinsClick(Sender: TObject);
    procedure SecsClick(Sender: TObject);
    procedure Exit2Click(Sender: TObject);
    procedure MenuHintClick(Sender: TObject);
    procedure ShowPerchIndicators1Click(Sender: TObject);

  private
    { Private declarations }
  public

    { Public declarations }
  end;

var
str1,  str2, str3, str4, str5, str6, str7, str8:string ;
{for Balance side of things, plus common ones}
togglie:boolean;
```

```
145: DayX, DayY: integer;
146: ddateA:string;
147: dayflag, initiate: boolean;
148:  Form1: TForm1;
149:   GlobalTime, t1, t2, t3, t4, t5, t6, t7, t8, TimeMarker,
150:  time11, time12, time13, time14, time21, time22, time23, time24, time31,
      time32,
151:  time33, time34, time41, time42, time43, time44, time51, time52, time53,
      time54,
152:  time61, time62, time63, time64, time71, time72, time73, time74, time81,
      time82,
153:  time83, time84, dDate, Time1b, Time2b, Time3b, Time4b, Time5b,
154:  Time6b, Time7b, Time8b,
155:  t1b, t2b, t3b, t4b, t5b, t6b, t7b, t8b:string;
156:   i1, i2, i3, i4, i5, i6, i7, i8,
157:   j1, j2, j3, j4, j5, j6, j7, j8,
158:   k1, k2, k3, k4, k5, k6, k7, k8,  LogPeriod,
159:   On1, On2, On3, On4, On5, On6, On7, On8, daycorrection, loggingcaptioncounter,
160:   logperiodDefiner, ValueBirdOnOff :integer;
161:  TimeInterval: extended;{changed from Extended Type}
162:   FeedTBData:Textfile;
163:   Flag1, Flag2, Flag3, Flag4, Flag5, Flag6, Flag7, Flag8,
164:   f1, f2, f3, f4, f5, f6, f7, f8, toggle,LoggerFlag,  Initialise, Flag,
165:   loggerflagadditional, PerchIndicatorFlag, ToggleFlag,
166:   f1b, f2b, f3b, f4b, f5b, f6b, f7b, f8b: Boolean;
167:
168:  {Variables associated with Perch Logger Operation}
169:   i, m1, m2, m3, m4, m5, m6, m7, m8, mb1, mb2, mb3, mb4, mb5, mb6, mb7, mb8,
170:   Jam1, Jam2, Jam3, Jam4, Jam5, Jam6, Jam7, Jam8, Count1, Count2, Count3,
      Count4,
171:   Count5, Count6, Count7, Count8,
172:   et1, et2, et3, et4, et5, et6, et7, et8 : integer;
173:   data1, data2, data3, data4, data5:longint;
174:   Pstr1, Pstr2, Pstr3, Pstr4, Pstr5,
175:   StrPerch1, StrPerch2, StrPerch3, StrPerch4, StrPerch5, StrPerch6,
176:   StrPerch7, StrPerch8, Bird1Pos, Bird2Pos, Bird3Pos, Bird4Pos, Bird5Pos,
177:   Bird6Pos, Bird7Pos, Bird8Pos:string;
178:   a, b, c, d, e, f, g, h, PerchTBData,
179:   Feed1, Feed2, Feed3, Feed4, Feed5, Feed6, Feed7, Feed8, Bal1, Bal2, Bal3,
      Bal4,
180:   Bal5, Bal6, Bal7, Bal8 :textfile;
181:   f1p, f2p, f3p, f4p, f5p, f6p, f7p, f8p, f9p, f10p, f11p, f12p, f13p, f14p,
      f15p, f16p, f17p,
182:   f18p, f19p, f20p, f21p, f22p, f23p, f24p, f25p, f26p, f27p, f28p, f29p, f30p,
      f31p, f32p:Boolean;
183:   t1p, t2p, t3p, t4p, t5p, t6p, t7p, t8p, t9p, t10p, t11p, t12p, t13p, t14p,
      t15p, t16p, t17p,
184:   t18p, t19p, t20p, t21p, t22p, t23p, t24p, t25p, t26p, t27p, t28p, t29p, t30p,
      t31p, t32p:string;
185:   On1p, On2p, On3p, On4p, On5p, On6p, On7p, On8p, On9p, On10p,
186:   On11p, On12p, On13p, On14p, On15p, On16p, On17p, On18p, On19p, On20p,
187:   On21p, On22p, On23p, On24p, On25p, On26p, On27p, On28p, On29p, On30p, On31p,
      On32p,
188:   OnPerch1, OnPerch2, OnPerch3, OnPerch4, OnPerch5, OnPerch6,
189:   OnPerch7, OnPerch8, On1b, On2b, On3b, On4b, On5b, On6b, On7b, On8b,
190:   PExt1, PExt2, PExt3, PExt4, PExt5, PExt6, PExt7, PExt8,
191:   Tare1, Tare2, Tare3, Tare4, Tare5, Tare6, Tare7, Tare8,
192:   PExt1a, PExt2a, PExt3a, PExt4a, PExt5a, PExt6a, PExt7a, PExt8a:single;
193:   Bird1, Bird2, Bird3, Bird4, Bird5, Bird6, Bird7, Bird8,
194:   PreBalOut1, PreBalOut2, PreBalOut3, PreBalOut4, PreBalOut5, PreBalOut6,
      PreBalOut7,
195:   PreBalOut8, BalOut1, BalOut2, BalOut3, BalOut4, BalOut5, BalOut6, BalOut7,
      BalOut8: String;
196: implementation
197:
198: uses Unit1aboutbox;
199:
200: {$R *.DFM}
201:
202: Function registerBoard(model, ba, irq:integer):integer;stdcall;external
      'Dio_tc.dll';
203: Function freeBoard(h:integer):integer;stdcall;external 'Dio_tc.dll';
```

```
204: Function DIOsetMode(h, chip, port, isInput:integer):integer;stdcall;external
      'Dio_tc.dll';
205: Function DIOsetChanWidth(h, chip, numBits:integer):integer;stdcall;external
      'Dio_tc.dll';
206: Function DIOgetData(h, chip, chan:integer;
      dat:plongint):integer;stdcall;external 'Dio_tc.dll';
207:
208:
209: function timetointegerMs(ATime:TdateTime): longint;
210: var
211: hours, minutes, seconds, milliseconds :word;
212: hours1:longint;
213: begin
214: decodetime (Atime, Hours, minutes, seconds, milliseconds);
215: hours1:=hours;
216: TimetoIntegerMs:=hours1*3600000+minutes*60000+seconds*1000+milliseconds;
217: end;
218:
219: Function DecToBin(dec:longint):string;
220: var x:integer;
221: begin
222: x:=200000000;
223: if dec-128>=0 then begin dec:=dec-128; x:=x+10000000; end;
224: if dec-64>=0 then begin dec:=dec-64; x:=x+1000000;end;
225: if dec-32>=0 then begin dec:=dec-32; x:=x+100000;end;
226: if dec-16>=0 then begin dec:=dec-16; x:=x+10000;end;
227: if dec-8>=0 then begin dec:=dec-8; x:=x+1000;end;
228: if dec-4>=0 then begin dec:=dec-4; x:=x+100;end;
229: if dec-2>=0 then begin dec:=dec-2; x:=x+10;end;
230: if dec-1>=0   then begin dec:=dec-1; x:=x+1;end;
231: result:=inttostr(x)
232: end;
233:
234: {BEGINNING OF COMM EVENT HANDLERS}
235: procedure TForm1.comm1ReceiveData(Buffer: Pointer; BufferLength: Word);
236: begin
237:  if GetTickCount>et1+4000 then edit1log.text:='--.-g';
238:   setlength(str1, bufferlength);
239:    Move(Buffer^, str1[1], BufferLength);
240:    PreBalOut1:=BalOut1;
241:     BalOut1:=concat(str1[1]+str1[7]+str1[8]+str1[9]+str1[10]);
242:      edit1.text:=BalOut1;
243:       if BalOut1=PreBalOut1 then Count1:=Count1+1
244:        else Count1:=0;
245: if Count1>=8 then begin
246:  PExt1a:=PExt1;
247:   PExt1:=strToFloat(concat(str1[7]+str1[8]+str1[9]+str1[10]));
248:    if str1[1]='-' then PExt1:=PExt1*-1;
249:     if PExt1<5 then if PExt1>-5 then Tare1:=PExt1;
250:       edit1tare.text:=FloatToStrF(Tare1, ffFixed, 3, 1);
251: if PExt1-Tare1>=10 then begin
252:  Bird1:=FloattoStrF(PExt1-Tare1, ffFixed, 3, 1);
253:   Append(Ball1);
254:    if et1+15000<gettickcount then PExt1a:=0;// 15 seconds after last write do
      this
255:     if Pext1a<>Pext1  then  begin
256:      Edit1log.text:=Bird1+' g'; et1:=gettickcount;
257:       WriteLn(Ball1, GlobalTime+', '+Bird1+', '+TimetoStr(Time)); end;
258:        CloseFile(Ball1); Count1:=-1;
259:         end;
260: end;
261:  end;//of procedure
262:
263: procedure TForm1.comm2ReceiveData(Buffer: Pointer; BufferLength: Word);
264: var
265: tareLimit:single;
266: begin
267: TareLimit:=6;
268: if GetTickCount>et2+4000 then edit2log.text:='--.-g';
269:   setlength(str2, bufferlength);
270:    Move(Buffer^, str2[1], BufferLength);
271:    PreBalOut2:=BalOut2;
```

```
272:     BalOut2:=concat(str2[1]+str2[7]+str2[8]+str2[9]+str2[10]);
273:      edit2.text:=BalOut2;
274:       if BalOut2=PreBalOut2 then Count2:=Count2+1
275:        else Count2:=0;
276: if Count2>=8 then begin
277:  PExt2a:=PExt2;
278:   PExt2:=strToFloat(concat(str2[7]+str2[8]+str2[9]+str2[10]));
279:    if str2[1]='-' then PExt2:=PExt2*-1;
280:     if PExt2<TareLimit then if PExt2>-TareLimit then Tare2:=PExt2;
281:       edit2tare.text:=FloatToStrF(Tare2, ffFixed, 3, 1);
282: if PExt2-Tare2>=10 then begin
283:  Bird2:=FloattoStrF(PExt2-Tare2, ffFixed, 3, 1);
284:   Append(Bal2);
285:    if et2+15000<gettickcount then PExt2a:=0;
286:     if Pext2a<>Pext2  then  begin
287:      Edit2log.text:=Bird2+' g'; et2:=gettickcount;
288:       WriteLn(Bal2, GlobalTime+', '+Bird2+', '+TimetoStr(Time)); end;
289:        CloseFile(Bal2); Count2:=-1;
290:         end;
291: end;
292:  end;//of procedure
293:
294: procedure TForm1.comm3ReceiveData(Buffer: Pointer; BufferLength: Word);
295: begin
296:  if GetTickCount>et3+4000 then edit3log.text:='--.-g';
297:   setlength(str3, bufferlength);
298:    Move(Buffer^, str3[1], BufferLength);
299:    PreBalOut3:=BalOut3;
300:     BalOut3:=concat(str3[1]+str3[7]+str3[8]+str3[9]+str3[10]);
301:      edit3.text:=BalOut3;
302:       if BalOut3=PreBalOut3 then Count3:=Count3+1
303:        else Count3:=0;
304: if Count3>=8 then begin
305:  PExt3a:=PExt3;
306:   PExt3:=strToFloat(concat(str3[7]+str3[8]+str3[9]+str3[10]));
307:    if str3[1]='-' then PExt3:=PExt3*-1;
308:     if PExt3<5 then if PExt3>-5 then Tare3:=PExt3;
309:       edit3tare.text:=FloatToStrF(Tare3, ffFixed, 3, 1);
310: if PExt3-Tare3>=10 then begin
311:  Bird3:=FloattoStrF(PExt3-Tare3, ffFixed, 3, 1);
312:   Append(Bal3);
313:    if et3+15000<gettickcount then PExt3a:=0;
314:     if Pext3a<>Pext3  then  begin
315:      Edit3log.text:=Bird3+' g'; et3:=gettickcount;
316:       WriteLn(Bal3, GlobalTime+', '+Bird3+', '+TimetoStr(Time)); end;
317:        CloseFile(Bal3); Count3:=-1;
318:         end;
319: end;
320:  end;//of procedure
321:
322:
323: procedure TForm1.comm4ReceiveData(Buffer: Pointer; BufferLength: Word);
324: begin
325:  if GetTickCount>et4+4000 then edit4log.text:='--.-g';
326:   setlength(str4, bufferlength);
327:    Move(Buffer^, str4[1], BufferLength);
328:    PreBalOut4:=BalOut4;
329:     BalOut4:=concat(str4[1]+str4[7]+str4[8]+str4[9]+str4[10]);
330:      edit4.text:=BalOut4;
331:       if BalOut4=PreBalOut4 then Count4:=Count4+1
332:        else Count4:=0;
333: if Count4>=8 then begin
334:  PExt4a:=PExt4;
335:   PExt4:=strToFloat(concat(str4[7]+str4[8]+str4[9]+str4[10]));
336:    if str4[1]='-' then PExt4:=PExt4*-1;
337:     if PExt4<5 then if PExt4>-5 then Tare4:=PExt4;
338:       edit4tare.text:=FloatToStrF(Tare4, ffFixed, 3, 1);
339: if PExt4-Tare4>=10 then begin
340:  Bird4:=FloattoStrF(PExt4-Tare4, ffFixed, 3, 1);
341:   Append(Bal4);
342:    if et4+15000<gettickcount then PExt4a:=0;
343:     if Pext4a<>Pext4  then  begin
```

```
344:      Edit4log.text:=Bird4+' g'; et4:=gettickcount;
345:       WriteLn(Bal4, GlobalTime+', '+Bird4+', '+TimetoStr(Time)); end;
346:        CloseFile(Bal4); Count4:=-1;
347:         end;
348: end;
349:  end;//of procedure
350:
351:
352: procedure TForm1.comm5ReceiveData(Buffer: Pointer; BufferLength: Word);
353: begin
354:  if GetTickCount>et5+4000 then edit5log.text:='--.-g';
355:   setlength(str5, bufferlength);
356:    Move(Buffer^, str5[1], BufferLength);
357:    PreBalOut5:=BalOut5;
358:     BalOut5:=concat(str5[1]+str5[7]+str5[8]+str5[9]+str5[10]);
359:      edit5.text:=BalOut5;
360:       if BalOut5=PreBalOut5 then Count5:=Count5+1
361:        else Count5:=0;
362: if Count5>=8 then begin
363:  PExt5a:=PExt5;
364:   PExt5:=strToFloat(concat(str5[7]+str5[8]+str5[9]+str5[10]));
365:    if str5[1]='-' then PExt5:=PExt5*-1;
366:     if PExt5<5 then if PExt5>-5 then Tare5:=PExt5;
367:       edit5tare.text:=FloatToStrF(Tare5, ffFixed, 3, 1);
368: if PExt5-Tare5>=10 then begin
369:  Bird5:=FloattoStrF(PExt5-Tare5, ffFixed, 3, 1);
370:   Append(Bal5);
371:    if et5+15000<gettickcount then PExt5a:=0;
372:     if Pext5a<>Pext5  then  begin
373:      Edit5log.text:=Bird5+' g'; et5:=gettickcount;
374:       WriteLn(Bal5, GlobalTime+', '+Bird5+', '+TimetoStr(Time)); end;
375:        CloseFile(Bal5); Count5:=-1;
376:         end;
377: end;
378:  end;//of procedure
379:
380:
381: procedure TForm1.comm6ReceiveData(Buffer: Pointer; BufferLength: Word);
382: begin
383:  if GetTickCount>et6+4000 then edit6log.text:='--.-g';
384:   setlength(str6, bufferlength);
385:    Move(Buffer^, str6[1], BufferLength);
386:    PreBalOut6:=BalOut6;
387:     BalOut6:=concat(str6[1]+str6[7]+str6[8]+str6[9]+str6[10]);
388:      edit6.text:=BalOut6;
389:       if BalOut6=PreBalOut6 then Count6:=Count6+1
390:        else Count6:=0;
391: if Count6>=8 then begin
392:  PExt6a:=PExt6;
393:   PExt6:=strToFloat(concat(str6[7]+str6[8]+str6[9]+str6[10]));
394:    if str6[1]='-' then PExt6:=PExt6*-1;
395:     if PExt6<5 then if PExt6>-5 then Tare6:=PExt6;
396:       edit6tare.text:=FloatToStrF(Tare6, ffFixed, 3, 1);
397: if PExt6-Tare6>=10 then begin
398:  Bird6:=FloattoStrF(PExt6-Tare6, ffFixed, 3, 1);
399:   Append(Bal6);
400:    if et6+15000<gettickcount then PExt6a:=0;
401:     if Pext6a<>Pext6  then  begin
402:      Edit6log.text:=Bird6+' g'; et6:=gettickcount;
403:       WriteLn(Bal6, GlobalTime+', '+Bird6+', '+TimetoStr(Time)); end;
404:        CloseFile(Bal6); Count6:=-1;
405:         end;
406: end;
407:  end;//of procedure
408:
409:
410: procedure TForm1.comm7ReceiveData(Buffer: Pointer; BufferLength: Word);
411: begin
412:  if GetTickCount>et7+4000 then edit7log.text:='--.-g';
413:   setlength(str7, bufferlength);
414:    Move(Buffer^, str7[1], BufferLength);
415:    PreBalOut7:=BalOut7;
```

```
416:     BalOut7:=concat(str7[1]+str7[7]+str7[8]+str7[9]+str7[10]);
417:      edit7.text:=BalOut7;
418:       if BalOut7=PreBalOut7 then Count7:=Count7+1
419:        else Count7:=0;
420: if Count7>=8 then begin
421:  PExt7a:=PExt7;
422:   PExt7:=strToFloat(concat(str7[7]+str7[8]+str7[9]+str7[10]));
423:    if str7[1]='-' then PExt7:=PExt7*-1;
424:     if PExt7<5 then if PExt7>-5 then Tare7:=PExt7;
425:       edit7tare.text:=FloatToStrF(Tare7, ffFixed, 3, 1);
426: if PExt7-Tare7>=10 then begin
427:  Bird7:=FloattoStrF(PExt7-Tare7, ffFixed, 3, 1);
428:   Append(Bal7);
429:    if et7+15000<gettickcount then PExt7a:=0;
430:     if Pext7a<>Pext7  then  begin
431:      Edit7log.text:=Bird7+' g'; et7:=gettickcount;
432:       WriteLn(Bal7, GlobalTime+', '+Bird7+', '+TimetoStr(Time)); end;
433:        CloseFile(Bal7); Count7:=-1;
434:         end;
435: end;
436:  end;//of procedure
437:
438:
439: procedure TForm1.comm8ReceiveData(Buffer: Pointer; BufferLength: Word);
440: begin
441:  if GetTickCount>et8+4000 then edit8log.text:='--.-g';
442:   setlength(str8, bufferlength);
443:    Move(Buffer^, str8[1], BufferLength);
444:    PreBalOut8:=BalOut8;
445:     BalOut8:=concat(str8[1]+str8[7]+str8[8]+str8[9]+str8[10]);
446:      edit8.text:=BalOut8;
447:       if BalOut8=PreBalOut8 then Count8:=Count8+1
448:        else Count8:=0;
449: if Count8>=8 then begin
450:  PExt8a:=PExt8;
451:   PExt8:=strToFloat(concat(str8[7]+str8[8]+str8[9]+str8[10]));
452:    if str8[1]='-' then PExt8:=PExt8*-1;
453:     if PExt8<5 then if PExt8>-5 then Tare8:=PExt8;
454:       edit8tare.text:=FloatToStrF(Tare8, ffFixed, 3, 1);
455: if PExt8-Tare8>=10 then begin
456:  Bird8:=FloattoStrF(PExt8-Tare8, ffFixed, 3, 1);
457:   Append(Bal8);
458:    if et8+15000<gettickcount then PExt8a:=0;
459:     if Pext8a<>Pext8  then  begin
460:      Edit8log.text:=Bird8+' g'; et8:=gettickcount;
461:       WriteLn(Bal8, GlobalTime+', '+Bird8+', '+TimetoStr(Time)); end;
462:        CloseFile(Bal8); Count8:=-1;
463:         end;
464: end;
465:  end;//of procedure
466:
467:
468: {END OF COMM EVENT HANDLERS AND START OF START/STOP EVENTS}
469: procedure TForm1.OpenPortsClick(Sender: TObject);
470: begin
471: if togglie=false then begin
472: beep;
473: comm1.startcomm;
474: comm2.startcomm;
475: comm3.startcomm;
476: comm4.startcomm;
477: comm5.startcomm;
478: comm6.startcomm;
479: comm7.startcomm;
480: comm8.startcomm;
481: Label5.caption:=' Ports Open ';
482: Label5.color:=cllime;
483: label5.width:=121;
484: OpenPorts.caption:='Close Comm Ports';
485: shape1.brush.color:=clred;
486: end;
487: if togglie=true then begin
```

```
488: beep;
489: comm1.stopcomm;
490: comm2.stopcomm;
491: comm3.stopcomm;
492: comm4.stopcomm;
493: comm5.stopcomm;
494: comm6.stopcomm;
495: comm7.stopcomm;
496: comm8.stopcomm;
497: Label5.caption:=' Ports Closed ';
498: Label5.color:=clred;
499: label5.width:=121;
500: OpenPorts.caption:='Open Comm Ports';
501: shape1.brush.color:=cllime;
502: end;
503: if togglie=true then togglie:=false else togglie:=true;
504: end;
505:
506:
507: procedure TForm1.ExitClick(Sender: TObject);
508: begin
509: Close;
510: end;
511:
512: procedure TForm1.Timer1Timer(Sender: TObject);
513: begin
514: Ddate:=DateToStr(Date);
515: DdateA:=' '+'['+ddate[1]+ddate[2]+' '+ddate[4]+ddate[5]+'
      '+ddate[7]+ddate[8]+']';
516: DayX:=DayOfWeek(Date); {Current Day of Week
517: Day Y is Previous day of week}
518:
519: if dayx<>dayy {then if initiate=true }then begin {if date has just changed then
      reassign filenames
520: for both perch files and balance files}
521:   AssignFile(a, 'c:\PerchData\Perch1'+DdateA+'.txt');{creates text files}
522:   AssignFile(b, 'c:\PerchData\Perch2'+DdateA+'.txt');
523:   AssignFile(c, 'c:\PerchData\Perch3'+DdateA+'.txt');
524:   AssignFile(d, 'c:\PerchData\Perch4'+DdateA+'.txt');
525:   AssignFile(e, 'c:\PerchData\Perch5'+DdateA+'.txt');
526:   AssignFile(f, 'c:\PerchData\Perch6'+DdateA+'.txt');
527:   AssignFile(g, 'c:\PerchData\Perch7'+DdateA+'.txt');
528:   AssignFile(h, 'c:\PerchData\Perch8'+DdateA+'.txt');
529:   AssignFile(PerchTBData, 'c:\PerchData\PerchTBData'+DdateA+'.txt');
530:   if FileExists('c:\PerchData\Perch1'+DdateA+'.txt')=true then begin
531:   reset(a); reset(b); reset(c); reset(d);
532:   reset(e); reset(f); reset(g); reset(h); reset(PerchTBData); end else begin
533:   rewrite(a); rewrite(b); rewrite(c); rewrite(d); rewrite(e);
534:   rewrite(f); rewrite(g); rewrite(h); rewrite(PerchTBData); end;
535:
536:   AssignFile(FeedTBData, 'c:\FeedPerchData\FeedTBData'+DdateA+'.txt');
537:   AssignFile(Feed1, 'c:\FeedPerchData\FeedPerch1'+DdateA+'.txt');
538:    AssignFile(Feed2, 'c:\FeedPerchData\FeedPerch2'+DdateA+'.txt');
539:     AssignFile(Feed3, 'c:\FeedPerchData\FeedPerch3'+DdateA+'.txt');
540:      AssignFile(Feed4, 'c:\FeedPerchData\FeedPerch4'+DdateA+'.txt');
541:       AssignFile(Feed5, 'c:\FeedPerchData\FeedPerch5'+DdateA+'.txt');
542:        AssignFile(Feed6, 'c:\FeedPerchData\FeedPerch6'+DdateA+'.txt');
543:         AssignFile(Feed7, 'c:\FeedPerchData\FeedPerch7'+DdateA+'.txt');
544:          AssignFile(Feed8, 'c:\FeedPerchData\FeedPerch8'+DdateA+'.txt');
545:
546:   AssignFile(Bal1, 'c:\EveryBalanceOutput\Cage1'+DdateA+'.txt');
547:    AssignFile(Bal2, 'c:\EveryBalanceOutput\Cage2'+DdateA+'.txt');
548:     AssignFile(Bal3, 'c:\EveryBalanceOutput\Cage3'+DdateA+'.txt');
549:      AssignFile(Bal4, 'c:\EveryBalanceOutput\Cage4'+DdateA+'.txt');
550:       AssignFile(Bal5, 'c:\EveryBalanceOutput\Cage5'+DdateA+'.txt');
551:        AssignFile(Bal6, 'c:\EveryBalanceOutput\Cage6'+DdateA+'.txt');
552:         AssignFile(Bal7, 'c:\EveryBalanceOutput\Cage7'+DdateA+'.txt');
553:          AssignFile(Bal8, 'c:\EveryBalanceOutput\Cage8'+DdateA+'.txt');
554:
555:   if FileExists('c:\EveryBalanceOutput\Cage1'+DdateA+'.txt') then begin
556:    reset(bal1);  reset(bal2);  reset(bal3);  reset(bal4);
557:      reset(bal5);  reset(bal6);  reset(bal7);  reset(bal8);  end else begin
```

```
558:  rewrite(Bal1); rewrite(Bal2); rewrite(Bal3); rewrite(Bal4);
559:   rewrite(Bal5); rewrite(Bal6); rewrite(Bal7); rewrite(Bal8);  end;
560:
561:  if FileExists('c:\FeedPerchData\FeedPerch1'+DdateA+'.txt')=true then begin
562:   reset(FeedTBData); reset(feed1); reset(feed2); reset(feed3);
563:   reset(feed4); reset(feed5); reset(feed6); reset(feed7); reset(feed8); end
      else begin
564:    rewrite(FeedTBData); rewrite(feed1); rewrite(feed2); rewrite(feed3);
565:    rewrite(feed4); rewrite(feed5); rewrite(feed6); rewrite(feed7);
      rewrite(feed8);
566:    end;
567:
568:  CloseFile(a); CloseFile(b); CloseFile(c); CloseFile(d);
569: CloseFile(e); CloseFile(f); CloseFile(g); CloseFile(h);
570: CloseFile(PerchTBData); CloseFile(FeedTBData);
571: CloseFile(feed1); CloseFile(feed2); CloseFile(feed3); CloseFile(feed4);
572: CloseFile(feed5); CloseFile(feed6); CloseFile(feed7); CloseFile(feed8);
573:  CloseFile(Bal1); CloseFile(Bal2);  CloseFile(Bal3);  CloseFile(Bal4);
574:   CloseFile(Bal5);  CloseFile(Bal6);  CloseFile(Bal7);  CloseFile(Bal8);
575: end;
576:
577: if LoggingCaptionCounter<100 then inc(LoggingCaptionCounter); if
      LoggingCaptionCounter=100 then
578:  if loggerflag=true then begin Label21.caption:='Waiting to Log T.B.
      Data.....';
579: label21.width:=121;
580: Label21.color:=clred; end;
581: GlobalTime:=inttostr(TimeToIntegerMs(time));{sets globaltime}
582:
583: {************code for Perch Logger**************}
584: DIOgetData(i, 0, 0, @data1);
585: DIOgetData(i, 0, 1, @data2); {gets IO data from ports}
586: DIOgetData(i, 0, 2, @data3);
587: DIOgetData(i, 8, 0, @data4);
588: DIOgetData(i, 8, 1, @data5); //new code for feed perch
589: Pstr1:=DectoBin(data1); {converts IO data into a binary string}
590: Pstr2:=DectoBin(data2);
591: Pstr3:=DectoBin(data3);
592: Pstr4:=DectoBin(data4);
593: Pstr5:=DectoBin(data5);//new code for feed perch
594:
595: if loggerFlag=true then begin {samples all of the eight perch units
596: plus the eight feeding perches- provided that loggerflag
597: is set to true}
598: {Zero is on, one is off}
599:
600: if Pstr5[9]='0' then if f1b=false then begin f1b:=true;
      Time1b:=TimetoStr(time);
601: t1b:=GlobalTime; end; //code for feed perch one
602: if Pstr5[9]='1' then if f1b=true then begin f1b:=false; inc(mb1);
      Append(feed1);
603: if strtofloat(globalTime)>strtofloat(t1b) then //prevents writing to file at
      turn of day
604: WriteLn(feed1, t1b +', '+time1b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t1b)));
605: CloseFile(Feed1);
606: On1b:=On1b+strtofloat(globaltime)-strtofloat(t1b); end;
607:
608: if Pstr5[8]='0' then if f2b=false then begin f2b:=true;
      Time2b:=TimetoStr(time);
609: t2b:=GlobalTime; end; //code for feed perch two
610: if Pstr5[8]='1' then if f2b=true then begin f2b:=false; inc(mb2);
      Append(feed2);
611: if strtofloat(globalTime)>strtofloat(t2b) then //prevents writing to file at
      turn of day
612: WriteLn(feed2, t2b +', '+time2b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t2b)));
613: CloseFile(Feed2);
614: On2b:=On2b+strtofloat(globaltime)-strtofloat(t2b); end;
615:
616: if Pstr5[7]='0' then if f3b=false then begin f3b:=true;
      Time3b:=TimetoStr(time);
```

```
617: t3b:=GlobalTime; end; //code for feed perch three
618: if Pstr5[7]='1' then if f3b=true then begin f3b:=false; inc(mb3);
      Append(feed3);
619: if strtofloat(globalTime)>strtofloat(t3b) then //prevents writing to file at
      turn of day
620: WriteLn(feed3, t3b +', '+time3b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t3b)));
621: CloseFile(Feed3);
622: On3b:=On3b+strtofloat(globaltime)-strtofloat(t3b); end;
623:
624: if Pstr5[6]='0' then if f4b=false then begin f4b:=true;
      Time4b:=TimetoStr(time);
625: t4b:=GlobalTime; end; //code for feed perch four
626: if Pstr5[6]='1' then if f4b=true then begin f4b:=false; inc(mb4);
      Append(feed4);
627: if strtofloat(globalTime)>strtofloat(t4b) then //prevents writing to file at
      turn of day
628: WriteLn(feed4, t4b +', '+time4b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t4b)));
629: CloseFile(Feed4);
630: On4b:=On4b+strtofloat(globaltime)-strtofloat(t4b); end;
631:
632: if Pstr5[5]='0' then if f5b=false then begin f5b:=true;
      Time5b:=TimetoStr(time);
633: t5b:=GlobalTime; end; //code for feed perch five
634: if Pstr5[5]='1' then if f5b=true then begin f5b:=false; inc(mb5);
      Append(feed5);
635: if strtofloat(globalTime)>strtofloat(t5b) then //prevents writing to file at
      turn of day
636: WriteLn(feed5, t5b +', '+time5b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t5b)));
637: CloseFile(Feed5);
638: On5b:=On5b+strtofloat(globaltime)-strtofloat(t5b); end;
639:
640: if Pstr5[4]='0' then if f6b=false then begin f6b:=true;
      Time6b:=TimetoStr(time);
641: t6b:=GlobalTime; end; //code for feed perch six
642: if Pstr5[4]='1' then if f6b=true then begin f6b:=false; inc(mb6);
      Append(feed6);
643: if strtofloat(globalTime)>strtofloat(t6b) then //prevents writing to file at
      turn of day
644: WriteLn(feed6, t6b +', '+time6b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t6b)));
645: CloseFile(Feed6);
646: On6b:=On6b+strtofloat(globaltime)-strtofloat(t6b); end;
647:
648: if Pstr5[3]='0' then if f7b=false then begin f7b:=true;
      Time7b:=TimetoStr(time);
649: t7b:=GlobalTime; end; //code for feed perch seven
650: if Pstr5[3]='1' then if f7b=true then begin f7b:=false; inc(mb7);
      Append(feed7);
651: if strtofloat(globalTime)>strtofloat(t7b) then //prevents writing to file at
      turn of day
652: WriteLn(feed7, t7b +', '+time7b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t7b)));
653: CloseFile(Feed7);
654: On7b:=On7b+strtofloat(globaltime)-strtofloat(t7b); end;
655:
656: if Pstr5[2]='0' then if f8b=false then begin f8b:=true;
      Time8b:=TimetoStr(time);
657: t8b:=GlobalTime; end; //code for feed perch eight
658: if Pstr5[2]='1' then if f8b=true then begin f8b:=false; inc(mb8);
      Append(feed8);
659: if strtofloat(globalTime)>strtofloat(t8b) then //prevents writing to file at
      turn of day
660: WriteLn(feed8, t8b +', '+time8b+',
      '+floattostr(strtofloat(globalTime)-strtofloat(t8b)));
661: CloseFile(Feed8);
662: On8b:=On8b+strtofloat(globaltime)-strtofloat(t8b); end;
663:
664:
665: if Pstr1[9]='0'then if f1p=false then begin f1p:=true; Time11:=Timetostr(time);
```

```
666: t1p:=GlobalTime; end;{Perch1.1}
667: if Pstr1[9]='1' then if f1p=true then begin f1p:=false; inc(m1); Append(a);
668: if strtofloat(globaltime)>strtofloat(t1p) then  {prevents writing to file at
      turn of day}
669: WriteLn(a, '1, ' + t1p +', '+time11+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t1p)));
670: CloseFile(a);
671: On1p:=On1p+strtofloat(globaltime)-strtofloat(t1p);end;
672:
673: if Pstr1[8]='0'then if f2p=false then begin f2p:=true; Time12:=Timetostr(time);
674: t2p:=GlobalTime; end;{Perch1.2}
675: if Pstr1[8]='1' then if f2p=true then begin f2p:=false; inc(m1);Append(a);
676: if strtofloat(globaltime)>strtofloat(t2p) then  {prevents writing to file at
      turn of day}
677: WriteLn(a, '2, ' + t2p+', '+time12+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t2p)));CloseFile(a);
678: On2p:=On2p+strtofloat(globaltime)-strtofloat(t2p);end;
679:
680: if Pstr1[7]='0'then if f3p=false then begin f3p:=true; Time13:=Timetostr(time);
681: t3p:=GlobalTime; end; {Perch1.3}
682: if Pstr1[7]='1' then if f3p=true then begin f3p:=false; inc(m1); Append(a);
683: if strtofloat(globaltime)>strtofloat(t3p) then  {prevents writing to file at
      turn of day}
684: WriteLn(a, '3, ' + t3p+', ' +time13+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t3p)));CloseFile(a);
685: On3p:=On3p+strtofloat(globaltime)-strtofloat(t3p);end;
686:
687: if Pstr1[6]='0'then if f4p=false then begin f4p:=true; Time14:=Timetostr(time);
688: t4p:=GlobalTime; end;  {Perch1.4}
689: if Pstr1[6]='1' then if f4p=true then begin f4p:=false; inc(m1); Append(a);
690: if strtofloat(globaltime)>strtofloat(t4p) then  {prevents writing to file at
      turn of day}
691: WriteLn(a, '4, ' + t4p+', '+time14+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t4p)));CloseFile(a);
692: On4p:=On4p+strtofloat(globaltime)-strtofloat(t4p);end;
693:
694: if Pstr1[5]='0'then if f5p=false then begin f5p:=true; Time21:=Timetostr(time);
695: t5p:=GlobalTime; end; {Perch2.1}
696: if Pstr1[5]='1' then if f5p=true then begin f5p:=false; inc(m2);Append(b);
697: if strtofloat(globaltime)>strtofloat(t5p) then  {prevents writing to file at
      turn of day}
698: WriteLn(b, '1, ' + t5p+', '+time21+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t5p)));CloseFile(b);
699: On5p:=On5p+strtofloat(globaltime)-strtofloat(t5p); end;
700:
701: if Pstr1[4]='0'then if f6p=false then begin f6p:=true; Time22:=Timetostr(time);
702: t6p:=GlobalTime; end;  {Perch 2.2}
703: if Pstr1[4]='1' then if f6p=true then begin f6p:=false; inc(m2);Append(b);
704: if strtofloat(globaltime)>strtofloat(t6p) then  {prevents writing to file at
      turn of day}
705: WriteLn(b, '2, ' + t6p+', '+time22+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t6p)));CloseFile(b);
706: On6p:=On6p+strtofloat(globaltime)-strtofloat(t6p); end;
707:
708: if Pstr1[3]='0'then if f7p=false then begin f7p:=true; Time23:=Timetostr(time);
709: t7p:=GlobalTime; end;  {Perch 2.3}
710: if Pstr1[3]='1' then if f7p=true then begin f7p:=false; inc(m2); Append(b);
711: if strtofloat(globaltime)>strtofloat(t7p) then  {prevents writing to file at
      turn of day}
712: WriteLn(b, '3, ' + t7p+', '+time23+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t7p)));CloseFile(b);
713: On7p:=On7p+strtofloat(globaltime)-strtofloat(t7p);end;
714:
715: if Pstr1[2]='0'then if f8p=false then begin f8p:=true; Time24:=Timetostr(time);
716: t8p:=GlobalTime; end;  {Perch 2.4}
717: if Pstr1[2]='1' then if f8p=true then begin f8p:=false; inc(m2); Append(b);
718: if strtofloat(globaltime)>strtofloat(t8p) then  {prevents writing to file at
      turn of day}
719: WriteLn(b, '4, ' + t8p+', '+time24+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t8p)));CloseFile(b);
720: On8p:=On8p+strtofloat(globaltime)-strtofloat(t8p);end;
721:
```

```
722: if Pstr2[9]='0'then if f9p=false then begin f9p:=true; Time31:=Timetostr(time);
723: t9p:=GlobalTime; end;{Perch 3.1}
724: if Pstr2[9]='1' then if f9p=true then begin f9p:=false; inc(m3); Append(c);
725: if strtofloat(globaltime)>strtofloat(t9p) then  {prevents writing to file at
      turn of day}
726: WriteLn(c, '1, ' + t9p+', '+time31+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t9p)));CloseFile(c);
727: On9p:=On9p+strtofloat(globaltime)-strtofloat(t9p); end;
728:
729: if Pstr2[8]='0'then if f10p=false then begin f10p:=true;
      Time32:=Timetostr(time);
730: t10p:=GlobalTime; end;{Perch 3.2}
731: if Pstr2[8]='1' then if f10p=true then begin f10p:=false; inc(m3); Append(c);
732: if strtofloat(globaltime)>strtofloat(t10p) then  {prevents writing to file at
      turn of day}
733: WriteLn(c, '2, ' + t10p+', '+time32+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t10p)));CloseFile(c);
734: On10p:=On10p+strtofloat(globaltime)-strtofloat(t10p); end;
735:
736: if Pstr2[7]='0'then if f11p=false then begin f11p:=true;
      Time33:=Timetostr(time);
737: t11p:=GlobalTime; end; {Perch 3.3}
738: if Pstr2[7]='1' then if f11p=true then begin f11p:=false; inc(m3); Append(c);
739: if strtofloat(globaltime)>strtofloat(t11p) then  {prevents writing to file at
      turn of day}
740: WriteLn(c, '3, ' + t11p+', '+time33+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t11p)));CloseFile(c);
741: On11p:=On11p+strtofloat(globaltime)-strtofloat(t11p); end;
742:
743: if Pstr2[6]='0'then if f12p=false then begin f12p:=true;
      Time34:=Timetostr(time);
744: t12p:=GlobalTime; end;  {Perch3 .4}
745: if Pstr2[6]='1' then if f12p=true then begin f12p:=false; inc(m3);  Append(c);
746: if strtofloat(globaltime)>strtofloat(t12p) then  {prevents writing to file at
      turn of day}
747: WriteLn(c, '4, ' + t12p+', '+time34+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t12p)));CloseFile(c);
748: On12p:=On12p+strtofloat(globaltime)-strtofloat(t12p);end;
749:
750: if Pstr2[5]='0'then if f13p=false then begin f13p:=true;
      Time41:=Timetostr(time);
751: t13p:=GlobalTime; end; {Perch 4.1}
752: if Pstr2[5]='1' then if f13p=true then begin f13p:=false; inc(m4); Append(d);
753: if strtofloat(globaltime)>strtofloat(t13p) then  {prevents writing to file at
      turn of day}
754: WriteLn(d, '1, ' + t13p+', '+time41+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t13p)));CloseFile(d);
755: On13p:=On13p+strtofloat(globaltime)-strtofloat(t13p); end;
756:
757: if Pstr2[4]='0'then if f14p=false then begin f14p:=true;
      Time42:=Timetostr(time);
758: t14p:=GlobalTime; end;  {Perch 4.2}
759: if Pstr2[4]='1' then if f14p=true then begin f14p:=false; inc(m4); Append(d);
760: if strtofloat(globaltime)>strtofloat(t14p) then  {prevents writing to file at
      turn of day}
761: WriteLn(d, '2, ' + t14p+', '+time42+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t14p)));CloseFile(d);
762: On14p:=On14p+strtofloat(globaltime)-strtofloat(t14p); end;
763:
764: if Pstr2[3]='0'then if f15p=false then begin f15p:=true;
      Time43:=Timetostr(time);
765: t15p:=GlobalTime; end;  {Perch 4.3}
766: if Pstr2[3]='1' then if f15p=true then begin f15p:=false; inc(m4);Append(d);
767: if strtofloat(globaltime)>strtofloat(t15p) then  {prevents writing to file at
      turn of day}
768: WriteLn(d, '3, ' + t15p+', '+time43+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t15p)));CloseFile(d);
769: On15p:=On15p+strtofloat(globaltime)-strtofloat(t15p); end;
770:
771: if Pstr2[2]='0'then if f16p=false then begin f16p:=true;
      Time44:=Timetostr(time);
772: t16p:=GlobalTime; end;  {Perch 4.4}
```

```
773: if Pstr2[2]='1' then if f16p=true then begin f16p:=false; inc(m4);Append(d);
774: if strtofloat(globaltime)>strtofloat(t16p) then  {prevents writing to file at
      turn of day}
775: WriteLn(d, '4, ' + t16p+', '+time44+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t16p)));CloseFile(d);
776: On16p:=On16p+strtofloat(globaltime)-strtofloat(t16p);end;
777:
778: if Pstr3[9]='0'then if f17p=false then begin f17p:=true;
      Time51:=Timetostr(time);
779: t17p:=GlobalTime; end;{Perch 5.1}
780: if Pstr3[9]='1' then if f17p=true then begin f17p:=false; inc(m5); Append(e);
781: if strtofloat(globaltime)>strtofloat(t17p) then  {prevents writing to file at
      turn of day}
782: WriteLn(e, '1, ' + t17p +', '+time51+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t17p)));CloseFile(e);
783: On17p:=On17p+strtofloat(globaltime)-strtofloat(t17p); end;
784:
785: if Pstr3[8]='0'then if f18p=false then begin f18p:=true;
      Time52:=Timetostr(time);
786: t18p:=GlobalTime; end;{Perch 5.2}
787: if Pstr3[8]='1' then if f18p=true then begin f18p:=false; inc(m5);  Append(e);
788: if strtofloat(globaltime)>strtofloat(t18p) then  {prevents writing to file at
      turn of day}
789: WriteLn(e, '2, ' + t18p+', '+time52+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t18p)));CloseFile(e);
790: On18p:=On18p+strtofloat(globaltime)-strtofloat(t18p); end;
791:
792: if Pstr3[7]='0'then if f19p=false then begin f19p:=true;
      Time53:=Timetostr(time);
793: t19p:=GlobalTime; end; {Perch 5.3}
794: if Pstr3[7]='1' then if f19p=true then begin f19p:=false;  inc(m5); Append(e);
795: if strtofloat(globaltime)>strtofloat(t19p) then  {prevents writing to file at
      turn of day}
796: WriteLn(e, '3, ' + t19p+', '+time53+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t19p)));CloseFile(e);
797: On19p:=On19p+strtofloat(globaltime)-strtofloat(t19p); end;
798:
799: if Pstr3[6]='0'then if f20p=false then begin f20p:=true;
      Time54:=Timetostr(time);
800: t20p:=GlobalTime; end;  {Perch 5.4}
801: if Pstr3[6]='1' then if f20p=true then begin f20p:=false; inc(m5);  Append(e);
802: if strtofloat(globaltime)>strtofloat(t20p) then  {prevents writing to file at
      turn of day}
803: WriteLn(e, '4, ' + t20p+', '+time54+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t20p)));CloseFile(e);
804: On20p:=On20p+strtofloat(globaltime)-strtofloat(t20p);end;
805:
806: if Pstr3[5]='0'then if f21p=false then begin f21p:=true;
      Time61:=Timetostr(time);
807: t21p:=GlobalTime; end; {Perch 6.1}
808: if Pstr3[5]='1' then if f21p=true then begin f21p:=false;  inc(m6); Append(f);
809: if strtofloat(globaltime)>strtofloat(t21p) then  {prevents writing to file at
      turn of day}
810: WriteLn(f, '1, ' + t21p+', '+time61+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t21p)));CloseFile(f);
811: On21p:=On21p+strtofloat(globaltime)-strtofloat(t21p);end;
812:
813: if Pstr3[4]='0'then if f22p=false then begin f22p:=true;
      Time62:=Timetostr(time);
814: t22p:=GlobalTime; end;  {Perch 6.2}
815: if Pstr3[4]='1' then if f22p=true then begin f22p:=false;  inc(m6); Append(f);
816: if strtofloat(globaltime)>strtofloat(t22p) then  {prevents writing to file at
      turn of day}
817: WriteLn(f, '2, ' + t22p+', '+time62+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t22p)));CloseFile(f);
818: On22p:=On22p+strtofloat(globaltime)-strtofloat(t22p);end;
819:
820: if Pstr3[3]='0'then if f23p=false then begin f23p:=true;
      Time63:=Timetostr(time);
821: t23p:=GlobalTime; end;  {Perch 6.3}
822: if Pstr3[3]='1' then if f23p=true then begin f23p:=false; inc(m6); Append(f);
823: if strtofloat(globaltime)>strtofloat(t23p) then  {prevents writing to file at
```

```
      turn of day}
824: WriteLn(f, '3, ' + t23p+', '+time63+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t23p)));CloseFile(f);
825: On23p:=On23p+strtofloat(globaltime)-strtofloat(t23p);end;
826:
827: if Pstr3[2]='0'then if f24p=false then begin f24p:=true;
      Time64:=Timetostr(time);
828: t24p:=GlobalTime; end;  {Perch 6.4}
829: if Pstr3[2]='1' then if f24p=true then begin f24p:=false;  inc(m6); Append(f);
830: if strtofloat(globaltime)>strtofloat(t24p) then  {prevents writing to file at
      turn of day}
831: WriteLn(f, '4, ' + t24p+', '+time64+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t24p)));CloseFile(f);
832: On24p:=On24p+strtofloat(globaltime)-strtofloat(t24p);end;
833:
834: if Pstr4[9]='0'then if f25p=false then begin f25p:=true;
      Time71:=Timetostr(time);
835: t25p:=GlobalTime; end;{Perch 7.1}
836: if Pstr4[9]='1' then if f25p=true then begin f25p:=false; inc(m7);  Append(g);
837: if strtofloat(globaltime)>strtofloat(t25p) then  {prevents writing to file at
      turn of day}
838: WriteLn(g, '1, ' + t25p+', ' +time71+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t25p)));CloseFile(g);
839: On25p:=On25p+strtofloat(globaltime)-strtofloat(t25p);end;
840:
841: if Pstr4[8]='0'then if f26p=false then begin f26p:=true;
      Time72:=Timetostr(time);
842: t26p:=GlobalTime; end;{Perch 7.2}
843: if Pstr4[8]='1' then if f26p=true then begin f26p:=false; inc(m7);  Append(g);
844: if strtofloat(globaltime)>strtofloat(t26p) then  {prevents writing to file at
      turn of day}
845: WriteLn(g, '2, ' + t26p +', '+time72+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t26p)));CloseFile(g);
846: On26p:=On26p+strtofloat(globaltime)-strtofloat(t26p);end;
847:
848: if Pstr4[7]='0'then if f27p=false then begin f27p:=true;
      Time73:=Timetostr(time);
849: t27p:=GlobalTime; end; {Perch 7.3}
850: if Pstr4[7]='1' then if f27p=true then begin f27p:=false; inc(m7);  Append(g);
851: if strtofloat(globaltime)>strtofloat(t27p) then  {prevents writing to file at
      turn of day}
852: WriteLn(g, '3, ' + t27p+', '+time73+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t27p)));CloseFile(g);
853: On27p:=On27p+strtofloat(globaltime)-strtofloat(t27p);end;
854:
855: if Pstr4[6]='0'then if f28p=false then begin f28p:=true;
      Time74:=Timetostr(time);
856: t28p:=GlobalTime; end;  {Perch 7.4}
857: if Pstr4[6]='1' then if f28p=true then begin f28p:=false; inc(m7);  Append(g);
858: if strtofloat(globaltime)>strtofloat(t28p) then  {prevents writing to file at
      turn of day}
859: WriteLn(g, '4, ' + t28p+', '+time74+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t28p)));CloseFile(g);
860: On28p:=On28p+strtofloat(globaltime)-strtofloat(t28p);end;
861:
862: if Pstr4[5]='0'then if f29p=false then begin f29p:=true;
      Time81:=Timetostr(time);
863: t29p:=GlobalTime; end; {Perch 8.1}
864: if Pstr4[5]='1' then if f29p=true then begin f29p:=false; inc(m8); Append(h);
865: if strtofloat(globaltime)>strtofloat(t29p) then  {prevents writing to file at
      turn of day}
866: WriteLn(h, '1, ' + t29p+', '+time81+',
      '+floattostr(strtofloat(globaltime)-strtofloat(t29p)));CloseFile(h);
867: On29p:=On29p+strtofloat(globaltime)-strtofloat(t29p);end;
868:
869: if Pstr4[4]='0'then if f30p=false then begin f30p:=true;
      Time82:=Timetostr(time);
870: t30p:=GlobalTime; end;  {Perch 8.2}
871: if Pstr4[4]='1' then if f30p=true then begin f30p:=false;  inc(m8);Append(h);
872: if strtofloat(globaltime)>strtofloat(t30p) then  {prevents writing to file at
      turn of day}
873: WriteLn(h, '2, ' + t30p+', '+time82+',
```

```
      '+floattostr(strtofloat(globaltime)-strtofloat(t30p)));CloseFile(h);
  874: On30p:=On30p+strtofloat(globaltime)-strtofloat(t30p); end;
  875:
  876: if Pstr4[3]='0'then if f31p=false then begin f31p:=true;
       Time83:=Timetostr(time);
  877: t31p:=GlobalTime; end;  {Perch 8.3}
  878: if Pstr4[3]='1' then if f31p=true then begin f31p:=false; inc(m8); Append(h);
  879: if strtofloat(globaltime)>strtofloat(t31p) then  {prevents writing to file at
       turn of day}
  880: WriteLn(h, '3, ' + t31p+', '+time83+',
       '+floattostr(strtofloat(globaltime)-strtofloat(t31p)));CloseFile(h);
  881: On31p:=On31p+strtofloat(globaltime)-strtofloat(t31p);end;
  882:
  883: if Pstr4[2]='0'then if f32p=false then begin f32p:=true;
       Time84:=Timetostr(time);
  884: t32p:=GlobalTime; end;  {Perch 8.4}
  885: if Pstr4[2]='1' then if f32p=true then begin f32p:=false; inc(m8); Append(h);
  886: if strtofloat(globaltime)>strtofloat(t32p) then  {prevents writing to file at
       turn of day}
  887: WriteLn(h, '4, ' + t32p+', '+time84+',
       '+floattostr(strtofloat(globaltime)-strtofloat(t32p)));CloseFile(h);
  888: On32p:=On32p+strtofloat(globaltime)-strtofloat(t32p);end;
  889:
  890: label11.caption:=' '+Pstr1[9]+Pstr1[8]+Pstr1[7]+Pstr1[6]+' ';
  891: label12.caption:=' '+Pstr1[5]+Pstr1[4]+Pstr1[3]+Pstr1[2]+' ';
  892: label13.caption:=' '+Pstr2[9]+Pstr2[8]+Pstr2[7]+Pstr2[6]+' ';
  893: label14.caption:=' '+Pstr2[5]+Pstr2[4]+Pstr2[3]+Pstr2[2]+' ';
  894: label15.caption:=' '+Pstr3[9]+Pstr3[8]+Pstr3[7]+Pstr3[6]+' ';
  895: label16.caption:=' '+Pstr3[5]+Pstr3[4]+Pstr3[3]+Pstr3[2]+' ';
  896: label17.caption:=' '+Pstr4[9]+Pstr4[8]+Pstr4[7]+Pstr4[6]+' ';
  897: label18.caption:=' '+Pstr4[5]+Pstr4[4]+Pstr4[3]+Pstr4[2]+' ';
  898: //creates strings to indicate Display Perch Status
  899: StrPerch1:=Concat(Pstr1[9]+Pstr1[8]+Pstr1[7]+Pstr1[6]);
  900: StrPerch2:=Concat(Pstr1[5]+Pstr1[4]+Pstr1[3]+Pstr1[2]);
  901: StrPerch3:=Concat(Pstr2[9]+Pstr2[8]+Pstr2[7]+Pstr2[6]);
  902: StrPerch4:=Concat(Pstr2[5]+Pstr2[4]+Pstr2[3]+Pstr2[2]);
  903: StrPerch5:=Concat(Pstr3[9]+Pstr3[8]+Pstr3[7]+Pstr3[6]);
  904: StrPerch6:=Concat(Pstr3[5]+Pstr3[4]+Pstr3[3]+Pstr3[2]);
  905: StrPerch7:=Concat(Pstr4[9]+Pstr4[8]+Pstr4[7]+Pstr4[6]);
  906: StrPerch8:=Concat(Pstr4[5]+Pstr4[4]+Pstr4[3]+Pstr4[2]);
  907:
  908: if PerchIndicatorFlag=true then begin
  909: if Pstr1[9]='0' then form2.label1.color:=clred else
       form2.label1.color:=clyellow;
  910: if Pstr1[8]='0' then form2.label2.color:=clred else
       form2.label2.color:=clyellow;
  911: if Pstr1[7]='0' then form2.label3.color:=clred else
       form2.label3.color:=clyellow;
  912: if Pstr1[6]='0' then form2.label4.color:=clred else
       form2.label4.color:=clyellow;
  913: if Pstr1[5]='0' then form2.label5.color:=clred else
       form2.label5.color:=clyellow;
  914: if Pstr1[4]='0' then form2.label6.color:=clred else
       form2.label6.color:=clyellow;
  915: if Pstr1[3]='0' then form2.label7.color:=clred else
       form2.label7.color:=clyellow;
  916: if Pstr1[2]='0' then form2.label8.color:=clred else
       form2.label8.color:=clyellow;
  917: if Pstr2[9]='0' then form2.label9.color:=clred else
       form2.label9.color:=clyellow;
  918: if Pstr2[8]='0' then form2.label10.color:=clred else
       form2.label10.color:=clyellow;
  919: if Pstr2[7]='0' then form2.label11.color:=clred else
       form2.label11.color:=clyellow;
  920: if Pstr2[6]='0' then form2.label12.color:=clred else
       form2.label12.color:=clyellow;
  921: if Pstr2[5]='0' then form2.label13.color:=clred else
       form2.label13.color:=clyellow;
  922: if Pstr2[4]='0' then form2.label14.color:=clred else
       form2.label14.color:=clyellow;
  923: if Pstr2[3]='0' then form2.label15.color:=clred else
       form2.label15.color:=clyellow;
```

```
924: if Pstr2[2]='0' then form2.label16.color:=clred else
      form2.label16.color:=clyellow;
925: if Pstr3[9]='0' then form2.label17.color:=clred else
      form2.label17.color:=clyellow;
926: if Pstr3[8]='0' then form2.label18.color:=clred else
      form2.label18.color:=clyellow;
927: if Pstr3[7]='0' then form2.label19.color:=clred else
      form2.label19.color:=clyellow;
928: if Pstr3[6]='0' then form2.label20.color:=clred else
      form2.label20.color:=clyellow;
929: if Pstr3[5]='0' then form2.label21.color:=clred else
      form2.label21.color:=clyellow;
930: if Pstr3[4]='0' then form2.label22.color:=clred else
      form2.label22.color:=clyellow;
931: if Pstr3[3]='0' then form2.label23.color:=clred else
      form2.label23.color:=clyellow;
932: if Pstr3[2]='0' then form2.label24.color:=clred else
      form2.label24.color:=clyellow;
933: if Pstr4[9]='0' then form2.label25.color:=clred else
      form2.label25.color:=clyellow;
934: if Pstr4[8]='0' then form2.label26.color:=clred else
      form2.label26.color:=clyellow;
935: if Pstr4[7]='0' then form2.label27.color:=clred else
      form2.label27.color:=clyellow;
936: if Pstr4[6]='0' then form2.label28.color:=clred else
      form2.label28.color:=clyellow;
937: if Pstr4[5]='0' then form2.label29.color:=clred else
      form2.label29.color:=clyellow;
938: if Pstr4[4]='0' then form2.label30.color:=clred else
      form2.label30.color:=clyellow;
939: if Pstr4[3]='0' then form2.label31.color:=clred else
      form2.label31.color:=clyellow;
940: if Pstr4[2]='0' then form2.label32.color:=clred else
      form2.label32.color:=clyellow;
941:
942: end;
943: //this resides outwith previous statement such that these indicators operate
      regardless
944: //of whether or not form 2 is showing or hidden.
945: if Pstr5[9]='0' then label28.color:=clred else label28.color:=clwhite;
946: if Pstr5[8]='0' then label29.color:=clred else label29.color:=clwhite;
947: if Pstr5[7]='0' then label30.color:=clred else label30.color:=clwhite;
948: if Pstr5[6]='0' then label31.color:=clred else label31.color:=clwhite;
949: if Pstr5[5]='0' then label32.color:=clred else label32.color:=clwhite;
950: if Pstr5[4]='0' then label33.color:=clred else label33.color:=clwhite;
951: if Pstr5[3]='0' then label34.color:=clred else label34.color:=clwhite;
952: if Pstr5[2]='0' then label35.color:=clred else label35.color:=clwhite;
953:
954: end;
955:
956: {*************end of code for Perch Logger***********}
957: initiate:=true;  {to prevent day change code being read on startup}
958: Dayy:=dayofweek(Date);{to detect change of day}
959: end;
960:
961:
962: procedure TForm1.FormCreate(Sender: TObject);
963: begin
964: logperiodDefiner:=60;
965: label21.caption:=' Begin Logging? ';
966: label21.width:=121;
967: timer2.enabled:=false;
968: {for Perch Logger}
969: logperiod:=strtoint(edit20.text);
970:   i:=registerBoard(214, $300, 9); {registers board}
971: DIOsetChanWidth(i, 0, 8);
972: DIOsetChanWidth(i, 8, 8);
973: DIOsetMode(i, 0, 0, 1);
974: DIOsetMode(i, 0, 1, 1);
975: DIOsetMode(i, 0, 2, 1);
976: DIOsetMode(i, 0, 3, 1);
977: DIOsetMode(i, 8, 0, 1);
```

```
 978: {end of Form Create Commands for Perch Logger}
 979: {Start of commands for Balance Logger}
 980: label9.caption:='  Suspended  '; label9.color:=cllime; label9.width:=121;
 981: form1.height:=455 ;
 982: Form1.left:=0;
 983: form1.top:=0;
 984: form1.width:=722;
 985: end;
 986:
 987: procedure TForm1.CreateClearFiles1Click(Sender: TObject);
 988: begin
 989: if MessageDlg('Do you really want to clear these files, I mean, there might be
       all sort of useful data in there- you could be throwing away years of hard
       work. Is this what you really want?',
 990:    mtWarning, [mbYes, mbNo], 0) = mrYes then begin
 991:  AssignFile(a, 'c:\PerchData\Perch1'+DdateA+'.txt');{creates text files}
 992:   AssignFile(b, 'c:\PerchData\Perch2'+DdateA+'.txt');
 993:   AssignFile(c, 'c:\PerchData\Perch3'+DdateA+'.txt');
 994:   AssignFile(d, 'c:\PerchData\Perch4'+DdateA+'.txt');
 995:   AssignFile(e, 'c:\PerchData\Perch5'+DdateA+'.txt');
 996:   AssignFile(f, 'c:\PerchData\Perch6'+DdateA+'.txt');
 997:   AssignFile(g, 'c:\PerchData\Perch7'+DdateA+'.txt');
 998:   AssignFile(h, 'c:\PerchData\Perch8'+DdateA+'.txt');
 999:   AssignFile(PerchTBData, 'c:\PerchData\PerchTBData'+DdateA+'.txt');
1000:   Rewrite(a); Rewrite(b); Rewrite(c); Rewrite(d); Rewrite(e);
1001:   Rewrite(f); Rewrite(g); Rewrite(h); Rewrite(PerchTBData);
1002:
1003:   AssignFile(FeedTBData, 'c:\FeedPerchData\FeedTBData'+DdateA+'.txt');
1004:
1005:   AssignFile(Feed1, 'c:\FeedPerchData\FeedPerch1'+DdateA+'.txt');
1006:   AssignFile(Feed2, 'c:\FeedPerchData\FeedPerch2'+DdateA+'.txt');
1007:   AssignFile(Feed3, 'c:\FeedPerchData\FeedPerch3'+DdateA+'.txt');
1008:   AssignFile(Feed4, 'c:\FeedPerchData\FeedPerch4'+DdateA+'.txt');
1009:   AssignFile(Feed5, 'c:\FeedPerchData\FeedPerch5'+DdateA+'.txt');
1010:   AssignFile(Feed6, 'c:\FeedPerchData\FeedPerch6'+DdateA+'.txt');
1011:   AssignFile(Feed7, 'c:\FeedPerchData\FeedPerch7'+DdateA+'.txt');
1012:   AssignFile(Feed8, 'c:\FeedPerchData\FeedPerch8'+DdateA+'.txt');
1013:
1014:    Rewrite(Feed1); Rewrite(Feed2);  Rewrite(Feed3); Rewrite(Feed4);
1015:   Rewrite(Feed5); Rewrite(Feed6); Rewrite(Feed7); Rewrite(Feed8);
1016:   Rewrite(FeedTBData);
1017:
1018: AssignFile(Bal1, 'c:\EveryBalanceOutput\Cage1'+DdateA+'.txt');
1019:  AssignFile(Bal2, 'c:\EveryBalanceOutput\Cage2'+DdateA+'.txt');
1020:   AssignFile(Bal3, 'c:\EveryBalanceOutput\Cage3'+DdateA+'.txt');
1021:    AssignFile(Bal4, 'c:\EveryBalanceOutput\Cage4'+DdateA+'.txt');
1022:     AssignFile(Bal5, 'c:\EveryBalanceOutput\Cage5'+DdateA+'.txt');
1023:      AssignFile(Bal6, 'c:\EveryBalanceOutput\Cage6'+DdateA+'.txt');
1024:       AssignFile(Bal7, 'c:\EveryBalanceOutput\Cage7'+DdateA+'.txt');
1025:        AssignFile(Bal8, 'c:\EveryBalanceOutput\Cage8'+DdateA+'.txt');
1026: Rewrite(Bal1); Rewrite(Bal2); Rewrite(Bal3); Rewrite(Bal4);
1027:  Rewrite(Bal5); Rewrite(Bal6); Rewrite(Bal7); Rewrite(Bal8);
1028:
1029: CloseFile(Bal1); CloseFile(Bal2); CloseFile(Bal3); CloseFile(Bal4);
1030:  CloseFile(Bal5); CloseFile(Bal6); CloseFile(Bal7); CloseFile(Bal8);
1031:
1032:  CloseFile(a); CloseFile(b); CloseFile(c); CloseFile(d);
1033: CloseFile(e); CloseFile(f); CloseFile(g); CloseFile(h);
1034: CloseFile(feed1); CloseFile(feed2); CloseFile(feed3); CloseFile(feed4);
1035: CloseFile(feed5); CloseFile(feed6); CloseFile(feed7); CloseFile(feed8);
1036: CloseFile(PerchTBData);
1037: end;
1038: end;
1039:
1040: procedure TForm1.Exit1Click(Sender: TObject);
1041: begin
1042: if MessageDlg('Have you thought this through fully?',
1043:    mtWarning, [mbYes, mbNo], 0) = mrYes then
1044: Close;
1045: end;
1046:
1047:
```

```
1048: procedure TForm1.Timer2Timer(Sender: TObject);
1049: begin
1050: {f1p to f32p are the flags which denote that a perch has been occupied from one
       period
1051: to the next. On1p to On32p are the totals of time spent on perches. These are
       corrected if
1052: a bird has spent a full period on a perch by setting t1p to t32p to GlobalTime.
       OnPerch1 to
1053: OnPerch8 are the totals per cage for time spent on perches. It is these which
       are logged.}
1054:
1055: timer1.enabled:=false;
1056: {loggerflag is universal flag variable to enable/disable all logging
       operations}
1057: Label21.caption:='Logging Time Budget Data';
1058: label21.width:=121;
1059: Label21.color:=cllime; LoggingCaptionCounter:=0;
1060: {****************Code For Perch Logger********************}
1061: if f1p=true then begin On1p:=On1p+(strtoint(GlobalTime)-strtoint(t1p));
       t1p:=GlobalTime; end;
1062: if f2p=true then begin On2p:=On2p+(strtoint(GlobalTime)-strtoint(t2p));
       t2p:=GlobalTime; end;
1063: if f3p=true then begin On3p:=On3p+(strtoint(GlobalTime)-strtoint(t3p));
       t3p:=GlobalTime; end;
1064: if f4p=true then begin On4p:=On4p+(strtoint(GlobalTime)-strtoint(t4p));
       t4p:=GlobalTime; end;
1065: if f5p=true then begin On5p:=On5p+(strtoint(GlobalTime)-strtoint(t5p));
       t5p:=GlobalTime; end;
1066: if f6p=true then begin On6p:=On6p+(strtoint(GlobalTime)-strtoint(t6p));
       t6p:=GlobalTime; end;
1067: if f7p=true then begin On7p:=On7p+(strtoint(GlobalTime)-strtoint(t7p));
       t7p:=GlobalTime; end;
1068: if f8p=true then begin On8p:=On8p+(strtoint(GlobalTime)-strtoint(t8p));
       t8p:=GlobalTime; end;
1069: if f9p=true then begin On9p:=On9p+(strtoint(GlobalTime)-strtoint(t9p));
       t9p:=GlobalTime; end;
1070: if f10p=true then begin On10p:=On10p+(strtoint(GlobalTime)-strtoint(t10p));
       t10p:=GlobalTime; end;
1071: if f11p=true then begin On11p:=On11p+(strtoint(GlobalTime)-strtoint(t11p));
       t11p:=GlobalTime; end;
1072: if f12p=true then begin On12p:=On12p+(strtoint(GlobalTime)-strtoint(t12p));
       t12p:=GlobalTime; end;
1073: if f13p=true then begin On13p:=On13p+(strtoint(GlobalTime)-strtoint(t13p));
       t13p:=GlobalTime; end;
1074: if f14p=true then begin On14p:=On14p+(strtoint(GlobalTime)-strtoint(t14p));
       t14p:=GlobalTime; end;
1075: if f15p=true then begin On15p:=On15p+(strtoint(GlobalTime)-strtoint(t15p));
       t15p:=GlobalTime; end;
1076: if f16p=true then begin On16p:=On16p+(strtoint(GlobalTime)-strtoint(t16p));
       t16p:=GlobalTime; end;
1077: if f17p=true then begin On17p:=On17p+(strtoint(GlobalTime)-strtoint(t17p));
       t17p:=GlobalTime; end;
1078: if f18p=true then begin On18p:=On18p+(strtoint(GlobalTime)-strtoint(t18p));
       t18p:=GlobalTime; end;
1079: if f19p=true then begin On19p:=On19p+(strtoint(GlobalTime)-strtoint(t19p));
       t19p:=GlobalTime; end;
1080: if f20p=true then begin On20p:=On20p+(strtoint(GlobalTime)-strtoint(t20p));
       t20p:=GlobalTime; end;
1081: if f21p=true then begin On21p:=On21p+(strtoint(GlobalTime)-strtoint(t21p));
       t21p:=GlobalTime; end;
1082: if f22p=true then begin On22p:=On22p+(strtoint(GlobalTime)-strtoint(t22p));
       t22p:=GlobalTime; end;
1083: if f23p=true then begin On23p:=On23p+(strtoint(GlobalTime)-strtoint(t23p));
       t23p:=GlobalTime; end;
1084: if f24p=true then begin On24p:=On24p+(strtoint(GlobalTime)-strtoint(t24p));
       t24p:=GlobalTime; end;
1085: if f25p=true then begin On25p:=On25p+(strtoint(GlobalTime)-strtoint(t25p));
       t25p:=GlobalTime; end;
1086: if f26p=true then begin On26p:=On26p+(strtoint(GlobalTime)-strtoint(t26p));
       t26p:=GlobalTime; end;
1087: if f27p=true then begin On27p:=On27p+(strtoint(GlobalTime)-strtoint(t27p));
       t27p:=GlobalTime; end;
```

```
1088: if f28p=true then begin On28p:=On28p+(strtoint(GlobalTime)-strtoint(t28p));
       t28p:=GlobalTime; end;
1089: if f29p=true then begin On29p:=On29p+(strtoint(GlobalTime)-strtoint(t29p));
       t29p:=GlobalTime; end;
1090: if f30p=true then begin On30p:=On30p+(strtoint(GlobalTime)-strtoint(t30p));
       t30p:=GlobalTime; end;
1091: if f31p=true then begin On31p:=On31p+(strtoint(GlobalTime)-strtoint(t31p));
       t31p:=GlobalTime; end;
1092: if f32p=true then begin On32p:=On32p+(strtoint(GlobalTime)-strtoint(t32p));
       t32p:=GlobalTime; end;
1093: OnPerch1:=On1p+On2p+On3p+On4p; OnPerch2:=On5p+On6p+On7p+On8p;
       OnPerch3:=On9p+On10p+On11p+On12p;
1094: OnPerch4:=On13p+On14p+On15p+On16p; OnPerch5:=On17p+On18p+On19p+On20p;
1095: OnPerch6:=On21p+On22p+On23p+On24p; OnPerch7:=On25p+On26p+On27p+On28p;
1096: OnPerch8:=On29p+On30p+On31p+On32p;
1097: {****************End of Code for Perch Logger*****************}
1098:
1099: {This chunk of code allows for bird being on balance perch overlapping the time
       interval}
1100: if f1b=true then begin On1b:=On1b+(strtoint(GlobalTime)-strtoint(t1b));
       t1b:=globalTime; end;
1101: if f2b=true then begin On2b:=On2b+(strtoint(GlobalTime)-strtoint(t2b));
       t2b:=globalTime; end;
1102: if f3b=true then begin On3b:=On3b+(strtoint(GlobalTime)-strtoint(t3b));
       t3b:=globalTime; end;
1103: if f4b=true then begin On4b:=On4b+(strtoint(GlobalTime)-strtoint(t4b));
       t4b:=globalTime; end;
1104: if f5b=true then begin On5b:=On5b+(strtoint(GlobalTime)-strtoint(t5b));
       t5b:=globalTime; end;
1105: if f6b=true then begin On6b:=On6b+(strtoint(GlobalTime)-strtoint(t6b));
       t6b:=globalTime; end;
1106: if f7b=true then begin On7b:=On7b+(strtoint(GlobalTime)-strtoint(t7b));
       t7b:=globalTime; end;
1107: if f8b=true then begin On8b:=On8b+(strtoint(GlobalTime)-strtoint(t8b));
       t8b:=globalTime; end;
1108:
1109: {***************Code for Perch Logger**************}
1110: if StrToFloat(GlobalTime)>StrtoFloat(TimeMarker) then begin
1111:
1112: if loggerflag=true then begin
1113: TimeInterval:=strtofloat(GlobalTime)-strtofloat(TimeMarker);
1114: TimeMarker:=GlobalTime;
1115:
1116: Append(PerchTBData);
1117: WriteLn(PerchTBData, Timetostr(Time)+', '+GlobalTime+',
       '+floattostr(OnPerch1)+', '+
1118: FloatToStrF(OnPerch1/TimeInterval,ffFixed,5,3)+', '+inttostr(m1)+', '+
       GlobalTime+', '+floattostr(OnPerch2)+', '+
1119: floattostrF(OnPerch2/TimeInterval,ffFixed,5,3)+', '+inttostr(m2)+', '+
       GlobalTime+', '+floattostr(OnPerch3)+', '+
1120: floattostrF(OnPerch3/TimeInterval,ffFixed,5,3)+', '+inttostr(m3)+', '+
       GlobalTime+', '+floattostr(OnPerch4)+', '+
1121: floattostrF(OnPerch4/TimeInterval,ffFixed,5,3)+', '+inttostr(m4)+', '+
       GlobalTime+', '+floattostr(OnPerch5)+', '+
1122: floattostrF(OnPerch5/TimeInterval,ffFixed,5,3)+', '+inttostr(m5)+', '+
       GlobalTime+', '+floattostr(OnPerch6)+', '+
1123: floattostrF(OnPerch6/TimeInterval,ffFixed,5,3)+', '+inttostr(m6)+', '+
       GlobalTime+', '+floattostr(OnPerch7)+', '+
1124: floattostrF(OnPerch7/TimeInterval,ffFixed,5,3)+', '+inttostr(m7)+', '+
       GlobalTime+', '+floattostr(OnPerch8)+', '+
1125: floattostrF(OnPerch8/TimeInterval,ffFixed,5,3)+', '+inttostr(m8)+', '+' * ');
       Closefile(PerchTBData);
1126: {*********Code for balance logger function**********}
1127: Append(FeedTBData); WriteLn(FeedTBData, TimeTostr(Time)+', '+
1128: globaltime+', '+floattostr(On1b)+',
       '+floattostrF(On1b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb1)+', '+
1129: globaltime+', '+floattostr(On2b)+',
       '+floattostrF(On2b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb2)+', '+
1130: globaltime+', '+floattostr(On3b)+',
       '+floattostrF(On3b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb3)+', '+
1131: globaltime+', '+floattostr(On4b)+',
       '+floattostrF(On4b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb4)+', '+
```

```
1132: globaltime+', '+floattostr(On5b)+',
       '+floattostrF(On5b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb5)+', '+
1133: globaltime+', '+floattostr(On6b)+',
       '+floattostrF(On6b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb6)+', '+
1134: globaltime+', '+floattostr(On7b)+',
       '+floattostrF(On7b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb7)+', '+
1135: globaltime+', '+floattostr(On8b)+',
       '+floattostrF(On8b/TimeInterval,ffFixed,5,3)+', '+inttostr(mb8)+', '+'  *  ');
1136: closefile(FeedTBData);
1137: {***********end of code for balance logger function***********}
1138: end;
1139: end;
1140: {of 'if loggerflag=true exception}
1141: On1p:=0;On2p:=0; On3p:=0; On4p:=0; On5p:=0; On6p:=0; On7p:=0; On8p:=0;
1142: On9p:=0; On10p:=0; On11p:=0; On12p:=0; On13p:=0; On14p:=0; On15p:=0; On16p:=0;
       On17p:=0;
1143: On18p:=0; On19p:=0; On20p:=0; On21p:=0; On22p:=0; On23p:=0; On24p:=0; On25p:=0;
       On26p:=0;
1144: On27p:=0; On28p:=0; On29p:=0; On30p:=0; On31p:=0; On32p:=0;
1145: m1:=0; m2:=0; m3:=0; m4:=0; m5:=0; m6:=0; m7:=0; m8:=0;
1146: mb1:=0; mb2:=0; mb3:=0; mb4:=0; mb5:=0; mb6:=0; mb7:=0; mb8:=0;
1147: On1b:=0;On2b:=0; On3b:=0; On4b:=0; On5b:=0; On6b:=0; On7b:=0; On8b:=0;
1148: TimeMarker:=Globaltime;
1149: timer1.enabled:=true;
1150:
1151: end;
1152:
1153: Procedure Tform1.BtnSuspendClick(sender:Tobject);
1154: begin
1155: if toggle=False then begin toggle:=true; initialise:=false;
1156: loggerFlag:=true; loggerflagadditional:=true; timer2.enabled:=true;
1157: Timer2.interval:=Strtoint(Edit20.text)*1000*LogPeriodDefiner;
1158: TimeMarker:=GlobalTime; Label9.caption:='  Logging  ';
1159: label9.color:=cllime; label9.width:=121;
1160: OnPerch1:=0; OnPerch2:=0;
1161: OnPerch3:=0; OnPerch4:=0; OnPerch5:=0; OnPerch6:=0; OnPerch7:=0; OnPerch8:=0;
1162: m1:=0; m2:=0; m3:=0; m4:=0; m5:=0; m6:=0; m7:=0; m8:=0;
1163: mb1:=0; mb2:=0; mb3:=0; mb4:=0; mb5:=0; mb6:=0; mb7:=0; mb8:=0;
1164: On1b:=0;On2b:=0; On3b:=0; On4b:=0; On5b:=0; On6b:=0; On7b:=0; On8b:=0;
1165: t1p:=GlobalTime; t2p:=GlobalTime; t3p:=GlobalTime; t4p:=GlobalTime;
1166: t5p:=GlobalTime; t6p:=GlobalTime; t7p:=GlobalTime; t8p:=GlobalTime;
1167: t9p:=GlobalTime; t10p:=GlobalTime; t11p:=GlobalTime; t12p:=GlobalTime;
1168: t13p:=GlobalTime; t14p:=GlobalTime; t15p:=GlobalTime; t16p:=GlobalTime;
1169: t17p:=GlobalTime; t18p:=GlobalTime; t19p:=GlobalTime; t20p:=GlobalTime;
1170: t21p:=GlobalTime; t22p:=GlobalTime; t23p:=GlobalTime; t24p:=GlobalTime;
1171: t25p:=GlobalTime; t26p:=GlobalTime; t27p:=GlobalTime; t28p:=GlobalTime;
1172: t29p:=GlobalTime; t30p:=GlobalTime; t31p:=GlobalTime; t32p:=GlobalTime;
1173: t1:=GlobalTime; t2:=GlobalTime; t3:=GlobalTime; t4:=GlobalTime;
1174: t5:=GlobalTime; t6:=GlobalTime; t7:=GlobalTime; t8:=GlobalTime;
       TimeMarker:=Globaltime; end
1175: else if toggle=true then begin toggle:=false; loggerFlag:=false;
       loggerflagadditional:=false;
1176: label9.caption:='  Suspended  '; Label9.color:=clred; label9.width:=121;
1177: label21.color:=clGray;
1178: timer2.enabled:=false; end;
1179: end;
1180:
1181: procedure TForm1.UpDown2Click(Sender: TObject; Button: TUDBtnType);
1182: begin
1183: LogPeriod:=Strtoint(Edit20.text);
1184: end;
1185:
1186: procedure TForm1.About1Click(Sender: TObject);
1187: begin
1188: aboutbox.show;
1189: end;
1190:
1191: procedure TForm1.MinsClick(Sender: TObject);
1192: begin
1193: label23.caption:=' Minutes ';
1194: Mins.checked:=true;
1195: secs.checked:=false;
```

```
1196: LogPeriodDefiner:=60;
1197: Timer2.interval:=Strtoint(Edit20.text)*1000*LogPeriodDefiner;
1198: end;
1199:
1200: procedure TForm1.SecsClick(Sender: TObject);
1201: begin
1202: label23.caption:=' Seconds ';
1203: Secs.checked:=true;
1204: Mins.checked:=false;
1205: LogPeriodDefiner:=1;
1206: Timer2.interval:=Strtoint(Edit20.text)*1000*LogPeriodDefiner;
1207: end;
1208:
1209: procedure TForm1.Exit2Click(Sender: TObject);
1210: begin
1211: if MessageDlg('Are you sure you want to exit?',
1212:    mtWarning, [mbYes, mbNo], 0) = mrYes then
1213: Close;
1214: end;
1215:
1216: procedure TForm1.MenuHintClick(Sender: TObject);
1217: begin
1218: if MenuHint.Checked=false then begin
1219: MenuHint.checked:=true;
1220: Form1.ShowHint:=true;
1221: end
1222: else if MenuHint.Checked=true then begin
1223: MenuHint.checked:=False;
1224: Form1.ShowHint:=False;
1225: end;
1226: end;
1227:
1228: procedure TForm1.ShowPerchIndicators1Click(Sender: TObject);
1229: begin
1230: if toggleFlag=false then begin
1231: Form2.show;
1232: PerchIndicatorFlag:=true;
1233: ShowPerchIndicators1.caption:='Hide Perch Indicators';
1234: end;
1235: if toggleFlag=true then
1236: begin
1237: ShowPerchIndicators1.caption:='Show Perch Indicators';
1238: PerchIndicatorFlag:=false;
1239: Form2.hide;
1240: end;
1241: if toggleFlag=true then toggleFlag:=false else toggleFlag:=true;
1242: end;
1243:
1244: initialization
1245:
1246: valueBirdOnOff:=2;
1247: Pext1:=0; Pext2:=0; Pext3:=0; Pext4:=0; Pext5:=0; Pext6:=0; Pext7:=0; Pext8:=0;
1248: Pext1a:=0; Pext2a:=0; Pext3a:=0; Pext4a:=0; Pext5a:=0; Pext6a:=0; Pext7a:=0;
      Pext8a:=0;
1249: Count1:=0; Count2:=0; Count3:=0; Count4:=0;
1250: Count5:=0; Count6:=0; Count7:=0; Count8:=0;
1251: Tare1:=0; Tare2:=0; Tare3:=0; Tare4:=0;
1252: Tare5:=0; Tare6:=0; Tare7:=0; Tare8:=0;
1253: end.
```

Appendix 2

Full listing of programming code (in Delphi/ Object Pascal) for the **FileParse** program described in Chapter 2.

```
 1: unit ModUnit;
 2:
 3: interface
 4:
 5: uses
 6:   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
 7:   StdCtrls, ComCtrls, Inifiles, FileCtrl, WinTypes, ExtCtrls, Menus, Gauges,
    math;
 8:
 9: type
10:   TForm1 = class(TForm)
11:     Label1: TLabel;
12:     FileEdit1: TEdit;
13:     DriveComboBox1: TDriveComboBox;
14:     DirectoryListBox1: TDirectoryListBox;
15:     FileListBox1: TFileListBox;
16:     Edit5: TEdit;
17:     Panel1: TPanel;
18:     Edit6: TEdit;
19:     Edit7: TEdit;
20:     Label6: TLabel;
21:     Edit8: TEdit;
22:     Label7: TLabel;
23:     Panel2: TPanel;
24:     Edit4: TEdit;
25:     ButtonCombineFiles: TButton;
26:     Label5: TLabel;
27:     Panel3: TPanel;
28:     Label3: TLabel;
29:     Edit3: TEdit;
30:     Label4: TLabel;
31:     Button4: TButton;
32:     Panel4: TPanel;
33:     Edit2: TEdit;
34:     UpDown2: TUpDown;
35:     Edit1: TEdit;
36:     UpDown1: TUpDown;
37:     BtnSetTime: TButton;
38:     BtnSetupTimes: TButton;
39:     Label2: TLabel;
40:     Label8: TLabel;
41:     Panel5: TPanel;
42:     TransformPerchdata: TButton;
43:     ButtonAwardPerchScore: TButton;
44:     Button6: TButton;
45:     Button7: TButton;
46:     HopsPerPerchButton: TButton;
47:     Panel6: TPanel;
48:     BtnTransformFile: TButton;
49:     BtnMassGainPerInt: TButton;
50:     BtnBaldata: TButton;
51:     BtnCorrectDrift: TButton;
52:     BtnCorrectErrantMassFiles: TButton;
53:     RateOfMassGainButton: TButton;
54:     EditResults: TEdit;
55:     Label9: TLabel;
56:     Label10: TLabel;
57:     EditTotal: TEdit;
58:     Button1: TButton;
59:     Button2: TButton;
60:     Button3: TButton;
61:     Button5: TButton;
62:     Button8: TButton;
63:     Button9: TButton;
64:     ParseTemperatureFile: TButton;
65:     Button10: TButton;
66:     Edit9: TEdit;
67:     Edit10: TEdit;
68:     Button11: TButton;
69:     BtnMeanHopGap: TButton;
70:     Button12: TButton;
71:     procedure BtnSetTimeClick(Sender: TObject);
72:     procedure UpDown2Click(Sender: TObject; Button: TUDBtnType);
73:     procedure FormCreate(Sender: TObject);
```

```
74:     procedure BtnSetupTimesClick(Sender: TObject);
75:     procedure BtnTransformFileClick(Sender: TObject);
76:     procedure About1Click(Sender: TObject);
77:     procedure TransformPerchdataClick(Sender: TObject);
78:     procedure RateOfMassGainButtonClick(Sender: TObject);
79:     procedure BtnBaldataClick(Sender: TObject);
80:     procedure BtnCorrectErrantMassFilesClick(Sender: TObject);
81:     procedure BtnCorrectDriftClick(Sender: TObject);
82:     procedure ButtonCombineFilesClick(Sender: TObject);
83:     procedure ButtonAwardPerchScoreClick(Sender: TObject);
84:     procedure BtnMassGainPerIntClick(Sender: TObject);
85:     procedure Button4Click(Sender: TObject);
86:     procedure Button5Click(Sender: TObject);
87:     procedure Button6Click(Sender: TObject);
88:     procedure Button7Click(Sender: TObject);
89:     procedure HopsPerPerchButtonClick(Sender: TObject);
90:     procedure UpDown1Click(Sender: TObject; Button: TUDBtnType);
91:     procedure Button1Click(Sender: TObject);
92:     procedure Button2Click(Sender: TObject);
93:     procedure Button3Click(Sender: TObject);
94:     procedure Button8Click(Sender: TObject);
95:     procedure Button9Click(Sender: TObject);
96:     procedure Button10Click(Sender: TObject);
97:     procedure ParseTemperatureFileClick(Sender: TObject);
98:     procedure Button11Click(Sender: TObject);
99:     procedure BtnMeanHopGapClick(Sender: TObject);
100:    procedure Button12Click(Sender: TObject);
101:
102:  private
103:    { Private declarations }
104:  public
105:    { Public declarations }
106:  end;
107:
108: var  IntTime, Time, TimeA:longint;
109:     Commacount:integer; PositionBird, Repeats, realTime, hourOfDay
110:     :string; CommaGapA, CommaGapB, CommaGapC,
111:     mk3, mk4:integer;
112:   Form1: TForm1;
113:   Timesini: Tinifile;
114:   intervalMinutes, hours, minutes:integer;
115:   intervals, totalPerchMovements, count, RepeatsInt, CageNumber:integer;
116: MassStr, HourOfDayStr, hourStr, minStr, StrA:string;
117: results, f, Times, tempFile: textfile;
118: total,  k, strLength,  mk2, CommaGap, d1, d2, d3, d4, d5, IntervalCounter,
119: start, endd, Time1, Time2, Time3, Time4, Time5, Time6, Time7, Time8, Time9,
      Time10:integer;
120: Flag, IntervalCheck, PerchFlag, NewBirdMassFlag, InSeconds, SecondsFlag,
      initialise,
121: Finished : Boolean;
122: Mass, AverageMass, TotalMass, AveCounter,
123: Mass1, Mass2, Mass3, Mass4, Mass5, Mass6, Mass7, Mass8, Mass9, Mass10,
      difference,
124: TareValue, BirdMass, PreviousBirdMass, HourOfDayExt:extended;
125: str, timestr,PreviousMassStr, PreviousAverage, PreviousRealTime:string;
126: StringLength, i, mk1, PreviousTime, IntervalTimeCheck:integer;
127:  PreviousMass, DeltaMass, RateOfMassChange, DeltaTime:extended;
128: MeanMass, TotalRepeats,  repeatsExt, MassExt, StartingMass,
      EndingMass:extended;
129:
130:  j,  Starttt:integer;  PerchID, PreviousPerchID,  Duration :string;
131: CommaFlag, HopGap:longint;
132: totalDisplayValue, DisplayValue, RateOfMassGain: extended;
133: NewPerchScore:real;
134: TimeFromPreviousLine, CurrentTimeInterval, DifferenceA, DifferenceB,
      TimeFromCurrentLine,
135: StartingTime, EndingTime :Integer;
136:  MinuteStr:String;
137: const
138: sixty=60;
139:
140: implementation
141:
142: uses Unit3;
```

```
143:
144: //*****************************************************************************
145: //this function handles a line of a textfile and states whether or not the
     coded time is
146: //presented in seconds or milliseconds, returing a Boolean value to denote it.
147: Function IsTimeInSeconds(Str:String):Boolean;
148: var
149: StrTime, StrRealTime: string;
150: hrs, mins, secs, totalSeconds, StrLen: longint;
151: begin
152: mk1:=0; mk2:=0;
153: StrLen:=length(str);
154: for i:=1 to StrLen do begin
155:  if str[i]=',' then if mk1=0 then mk1:=i else if mk2=0 then mk2:=i;
156:                          end;//comma positions captured, up to two maximum!!
157:  for i:=1 to (mk1-1) do
158:   StrTime:=StrTime+str[i];
159:    IntTime:=StrToInt(StrTime);
160:           for i:=mk2+2 to mk2+9 do
161:            StrRealTime:=StrRealTime+Str[i];
162:
163: hrs:=strtoint(concat(StrRealTime[1]+StrRealTime[2]));
164: mins:=strtoint(concat(StrRealTime[4]+StrRealTime[5]));
165: secs:=strtoint(concat(StrRealTime[7]+StrRealTime[8]));
166: hrs:=hrs*3600;
167: mins:=mins*60;
168: TotalSeconds:=hrs+mins+secs;
169: if Inttime>TotalSeconds then result:=false else result:=true;
170: end;
171:
172: Function NextDayOfYear(PresentDay:string): string;
173: var
174: day, month, year : string;
175: begin
176: day:=presentDay[1]+presentDay[2];
177:  month:=presentDay[4]+presentDay[5];
178:   year:=presentDay[7]+presentDay[8];
179:    day:=IntToStr(StrToInt(day)+1);
180:     if Length(day)=1 then day:='0'+day;
181: if month='01' then if day='32' then begin day:='01'; month:='02'; end;
182:  if month='02' then if day='29' then begin day:='01'; month:='03'; end;
183:   if month='03' then if day='32' then begin day:='01'; month:='04'; end;
184:    if month='04' then if day='31' then begin day:='01'; month:='05'; end;
185:     if month='05' then if day='32' then begin day:='01'; month:='06'; end;
186:      if month='06' then if day='31' then begin day:='01'; month:='07'; end;
187:       if month='07' then if day='32' then begin day:='01'; month:='08'; end;
188:        if month='08' then if day='32' then begin day:='01'; month:='09'; end;
189:         if month='09' then if day='31' then begin day:='01'; month:='10'; end;
190:          if month='10' then if day='32' then begin day:='01'; month:='11'; end;
191:           if month='11' then if day='31' then begin day:='01'; month:='12';
     end;
192:            if month='12' then if day='32' then begin day:='01'; month:='01';
     end;
193: result:=day+' '+month+' '+year;
194: if presentDay='31 12 97' then result:='01 01 98';
195:  if presentDay='31 12 98' then result:='01 01 99';
196:   if presentDay='31 12 99' then result:='01 01 00';
197:    if presentDay='31 12 00' then result:='01 01 01';
198: end;
199:
200:
201:
202: {$R *.DFM}
203:
204: procedure TForm1.BtnSetTimeClick(Sender: TObject);
205: begin
206: if StrToInt(edit1.text)>(6) then MessageDlg('Value must be 6 or less',
     mtwarning,
207: [mbYes, mbNo], 0);
208: intervals:=strtoint(edit1.text);
209: {intervalMinutes:=Sixty/intervals;}
210: TimesIni.WriteInteger('TimeIntervals','Hour '+edit2.text, intervals);
211: edit2.text:=inttostr(strtoint(edit2.text)+1);
212: if strtoint(edit2.text)>23 then edit2.text:='23';
```

```
213: end;
214:
215:
216: procedure TForm1.UpDown2Click(Sender: TObject; Button: TUDBtnType);
217: begin
218: with Timesini do
219: edit1.text:=inttostr((Readinteger('TimeIntervals', 'Hour '+edit2.text, 0)));
220:
221: end;
222:
223: procedure TForm1.FormCreate(Sender: TObject);
224: begin
225: form1.top:=1; form1.left:=1;
226: TimesIni:=TiniFile.Create('TimeIntervals.Ini');
227: end;
228:
229: procedure TForm1.BtnSetupTimesClick(Sender: TObject);
230: var x,y:integer;
231: IntNumber:integer;
232: begin
233:      TimesIni.EraseSection('SecondGaps');
234: y:=0;// y is set to zero.
235: for x:=0 to 23 do begin
236: with timesini do IntNumber:=ReadInteger('TimeIntervals', 'Hour '+inttostr(x),
     0);
237: if IntNumber=1 then begin
238: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
239: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
     inc (y);
240: end;//of if IntNumber=1 routine
241: if IntNumber=2 then begin
242: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
243: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1799);
     inc (y);
244: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+1800);
245: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
     inc (y);
246: end;//of if IntNumber=2 routine
247: if IntNumber=3 then begin
248: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
249: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1199);
     inc (y);
250: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+1200);
251: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+2399);
     inc (y);
252: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+2400);
253: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
     inc (y);
254: end;//of if IntNumber=3 routine
255: if IntNumber=4 then begin
256: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
257: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+899);
     inc (y);
258: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+900);
259: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1799);
     inc (y);
260: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+1800);
261: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+2699);
     inc (y);
262: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
     3600*x+2700);
263: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
     inc (y);
264: end;//of if IntNumber=4 routine
265: if IntNumber=5 then begin
266: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
267: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+719);
     inc (y);
268: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
```

```
      3600*x+720);
269: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1439);
      inc (y);
270: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+1440);
271: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+2159);
      inc (y);
272: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+2160);
273: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+2879);
      inc (y);
274: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+2880);
275: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
      inc (y);
276: end;//of if IntNumber=5 routine
277: if IntNumber=6 then begin
278: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start', 3600*x);
279: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+599);
      inc (y);
280: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+600);
281: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1439);
      inc (y);
282: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+1199);
283: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1200);
      inc (y);
284: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+1799);
285: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+1800);
      inc (y);
286: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+2399);
287: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+2400);
      inc (y);
288: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'Start',
      3600*x+2999);
289: Timesini.WriteInteger('SecondGaps', 'interval'+inttostr(y)+'End', 3600*x+3599);
      inc (y);
290: end;//of if IntNumber=6 routine
291: end;
292: end;
293:
294: procedure TForm1.BtnTransformFileClick(Sender: TObject);
295:  var i, mk1, Starttt:integer; TimeStr, str:string;
296: CurrentDate:string;
297: label 1; label 2; label 3;
298: begin
299: //code highlighted below is responsible for reading files sequentially
300: ////////////////////////////////////////////////////////////////////////////
301: CurrentDate:=edit6.text;//sets current date
302:
303: 2:
304:  FileListBox1.color:=clwhite;filelistbox1.refresh;//flashes list box to denote
      cycling
305:  CageNumber:=CageNumber+1; if CageNumber=9 then //flips to next day and cage 1
306:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
307:  edit7.text:=label1.caption+'\'+'Cage '+IntToStr(CageNumber)+'
      ['+edit6.text+']';
308:   edit7.refresh;
309: edit6.text:=CurrentDate; edit6.refresh; //displays current date and updates
      display
310: if FileExists(label1.caption+'\'+'Cage'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt')
311:  then begin //if File is present in selected folder then run the following code
312:
313: assignFile(F,  label1.caption+'\'+'Cage'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt');
314: ////////////////////////////////////////////////////////////////////////////
315: edit5.text:='data to c:\ParsedMassData'; //indicates pathway to results file
316:  FileListBox1.color:=cllime;filelistbox1.refresh;//flashes the list box to
      denote end of cycle
317: btnTransformFile.Default:=true;//makes the button the default control
```

```
318: starttt:=0;
319:  Assignfile(TempFile, 'c:\ParsedMassData\Temp.txt');//defines the temporary
     file
320:   AssignFile(Times, 'c:\ParsedMassData\Current Time Divisions (Interval
     Beginning....).txt');
321: ///removed 20Feb 1999  Assignfile(Results, 'c:\ParsedMassData\Modi
     '+FileEdit1.text);
322: AssignFile(Results, 'c:\temp\ModiCage'+IntToStr(CageNumber)+'
     ['+CurrentDate+'].txt');
323:   //AssignFile(F, label1.caption+'\'+FileEdit1.text);
324:      reset(F);//resets the current data file
325:       rewrite(results); //clears current results file ready for writing
326:        Rewrite(Times);  //clears times file
327: //this section of code pre-reads the file and calculates the number of lines
328: readLn(F, Str);
329:  InSeconds:=IsTimeInSeconds(str);//function call to determine whether time is
     in secs or MSecs
330: while Str<>'' do begin Total:=Total+1;
331:    //EditTotal.text:=inttostr(total);
332:     //EditTotal.refresh;
333: ReadLn(F, Str); end;
334:
335: CloseFile(F); //************end of line counter- variable is called
     Total.******************
336:   Reset(F); //resets current data file again ready for reading through again
337: Edit3.text:=inttostr(total);//displays total number of lines in current data
     file
338:  Flag:=False;
339:   CommaCount:=1;
340:
341:
342: for k:=1 to total do begin// beginning of MAIN
     ROUTINE************************************
343:     mk1:=0; mk2:=0;
344:    readLn(F, Str); //reads line into Str variable
345:      StrLength:=Length(Str); //reads string length into 'Length' variable
346:   for i:=1 to StrLength do begin//finds commas, two of them
347: if str[i]=',' then if mk1=0 then mk1:=i else if mk2=0 then mk2:=i;
348:                             end;  //of comma finding routine, only finds two
     then stops
349:
350:
351:  CommaGapA:=mk2-mk1;
352:    CommaGap:=mk2-mk1; //calculates spacing between commas
353: d1:=mk1+2; d2:=mk1+3; d3:=mk1+4; d4:=mk1+5;//defines positions of digits 1 to 4
354: if CommaGap=3 then if MessageDlg('File has awfully dodgy data- do you wanna
     abort?', mtwarning,
355: [mbYes, mbNo], 0)=mrYes then close;//this shuts down the program if the data is
     corrupt
356: if CommaGap=4 then if str<>'' then MassStr:=concat(str[d1]+str[d2]);//for two
     digit mass
357: if CommaGap=6 then if str<>'' then
     MassStr:=Concat(str[d1]+str[d2]+str[d3]+str[d4]);//with four
358:  //note that masses can only have two or four significant digits
359:   mass:=strtoFloat(MassStr); //contains value of mass reading for current line
360:
361: if InSeconds=true then begin//runs InSeconds function to determine that time is
     not secs, not msecs
362:  if mk1=2 then if str<>'' then timeStr:=str[1];
363:   if mk1=3 then if str<>'' then timeStr:=concat(str[1]+str[2]);
364:    if mk1=4 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]);
365:     if mk1=5 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
366:      if mk1=6 then if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
367:                         end
368:                         else //if time is in milliseconds it extracts time using
     code below
369:                              begin
370:  if mk1=4 then if str<>'' then timeStr:=concat(str[1]);
371:  if mk1=5 then if str<>'' then timeStr:=concat(str[1]);
372:  if mk1=6 then if str<>'' then timeStr:=concat(str[1]);
373:  if mk1=7 then if str<>'' then timeStr:=concat(str[1]+str[2]);
374:  if mk1=8 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
375:  if mk1=9 then if str<>'' then
```

```
    timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
376:                          end;
377: Time:=strtoint(timeStr);//this is the time in seconds
378: //*************************************************************************
379: 1:
380:  with TimesIni do begin
381:       Start:=ReadInteger('SecondGaps',
     'Interval'+InttoStr(IntervalCounter)+'Start', -1);
382:       Endd:=ReadInteger('SecondGaps',
     'Interval'+InttoStr(IntervalCounter)+'End', -1);
383:                  end; //start and end times are read from 'ini' file into
     memory
384:
385:  if Time>=start then if Time<=endd then begin//if time is within current period
386:       TotalMass:=TotalMass+Mass; //adds current mass onto running total value
387:       AveCounter:=AveCounter+1;//increments this count to allow mean to be
     calculated
388:                                        end    else  begin
389:
390:     if AveCounter=0 then begin
391:       Append(Results);
392:       WriteLn(Results, 0);
393:       CloseFile(Results);
394:       IntervalCounter:=IntervalCounter+1;
395:       goto 1;
396:                         end      else
397:        begin //current time is outwith current period, therefore write stored
     mean to
398:              //textfile before moving on.
399:          Append(Results);
400:          WriteLn(Results, floattoStrF(TotalMass/AveCounter, ffFixed, 4, 1));
401:          PreviousAverage:=floattoStrF(TotalMass/AveCounter, ffFixed, 4, 1);
402:          CloseFile(Results);
403:          TotalMass:=0;  //resets total
404:          AveCounter:=0; //resets this counter
405:          IntervalCounter:=IntervalCounter+1;//clicks forward to next time
     interval
406:          goto 1; //sends execution point back to label 1
407:        end;                                              end;
408:
409: //*************************************************************************
410: end; //end of 'for k=1 to total do'
     routine*********************************************************
411:
412: intervalCounter:=IntervalCounter+1;//clicks forward to next time interval
413:
414:   if AveCounter>0 then begin
415:      Append(results);
416:      WriteLn(results, floattostrF(TotalMass/AveCounter, ffFixed, 4, 1));
417:      PreviousAverage:=floattoStrF(TotalMass/AveCounter, ffFixed, 4, 1);
418:      CloseFile(Results);
419:      TotalMass:=0;
420:      AveCounter:=0;
421:                        end;
422:
423:
424:   CloseFile(F);
425:           while start<>-1 do begin
426:
427:             with TimesIni do
428:             Start:=ReadInteger('SecondGaps',
     'Interval'+inttostr(IntervalCounter)+'Start', -1);
429:             IntervalCounter:=IntervalCounter+1;
430:             Append(Results);
431:             if start<>-1 then WriteLn(Results, 0);//writes zeros onto end of
     results if data
432:             //is absent
433:             CloseFile(Results);
434:                       end;
435:
436:
437:    IntervalCounter:=0;
438:  for i:=1 to 80 do begin //this routine writes the current time intervals file
439:  with TimesIni do
```

```
440: Start:=ReadInteger('SecondGaps', 'Interval'+inttostr(IntervalCounter)+'Start',
      -1);
441: hours:=start div 3600;
442: minutes:=(start-(hours*3600))div 60;
443: hourStr:=intToStr(Hours); if length(HourStr)=1 then
      HourStr:=concat('0'+HourStr);
444: MinStr:=intToStr(Minutes); if length(MinStr)=1 then MinStr:=concat(MinStr+'0');
445: TimeStr:=concat(hourStr+':'+minStr);
446: if start<>-1 then WriteLn(Times, (inttostr(start div 60))+', '+TimeStr);
447: intervalcounter:=intervalcounter+1;
448: end;
449:   reset(F);
450:    CloseFile(F);
451: mk1:=0; str:=''; TimeStr:='';
452: Flag:=false;
453: IntervalCheck:=false;
454:   CloseFile(Times);
455: Total:=0;
456:   IntervalCounter:=0;
457:   str:=''; //tidies up when finished
458:  mass:=0; intervalCounter:=0; aveCounter:=0; averageMass:=0; Start:=0; Endd:=0;
459:  TotalMass:=0; CommaCount:=1;
460:   PreviousAverage:='00';
461: FileListBox1.color:=clWindow;
462: ////////////////////////////
463:  end;//of 'if FileExist routine'**********************************
464:
465: if CurrentDate=edit8.text then goto 3;//if current date has hit end date then
      go to 3 and end
466: goto 2;
467: 3:
468: ////////////////////////////
469:  end;  //of mass file parsing routine
470:
471:
472:
473:
474:
475: procedure TForm1.About1Click(Sender: TObject);
476: begin
477: aboutbox.show;
478: end;
479:
480: procedure TForm1.TransformPerchdataClick(Sender: TObject);
481: var
482: i, mk1, Starttt :integer; TimeStr, str, CurrentDate:string;
483: label 1; label 2; label 3;
484: begin
485: CurrentDate:=edit6.text;//sets starting date using default value
486:
487: 2:
488:  FileListBox1.color:=clwhite;filelistbox1.refresh;
489:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
      back to 1 again
490:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
491:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
      ['+edit6.text+']';
492:   edit7.refresh;  //displays the file (with filepath) currently being parsed
      (if it exists)
493: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
494: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt')
495: //checks whether file exists or not, if so then the code below is executed
496:  then begin
497:
498: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt');
499: //F identifies the textfile currently being parsed
500: edit5.text:='data to c:\ParsedPerchData';
501: totalPerchMovements:=0; //this variable is used to count total hops per period
502:
503: TransformPerchData.default:=true;//makes button the new default control
504: AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
505: AssignFile(Results, 'c:\temp\ModiPerch'+IntToStr(CageNumber)+'
```

```
    ['+CurrentDate+'].txt');
506:  //assigns files for storage of time divisions and the Results Output (Modi
    prefix)
507:
508:   reset(F);        //resets file being read
509:   Rewrite(times);  //clears any previous content in times file
510:   rewrite(results);//clears any previous content in results file
511:
512:   //this section of code pre-reads the file and calculates the number of lines.
513:   readLn(F, Str);
514:   while Str<>'' do begin Total:=Total+1; //total lines
515:      //EditTotal.text:=inttostr(total);
516:     //EditTotal.refresh;
517:   ReadLn(F, Str);
518:                     end;
519: CloseFile(F); //end of line counter- no. of lines held in Total.
520:   Edit3.text:=inttostr(total); //displays total lines
521:   Reset(F);   //resets file being read again
522:   FileListBox1.color:=cllime;filelistbox1.refresh; //causes FileListBox to
    flash
523:   for k:=1 to total do begin //reads through each line of the textfile, this is
    the MAIN ROUTINE
524:      ReadLn(F, Str); //loads current line of textfile into Str variable.
525:      if k=1 then Str:='1, 10000, 00:00:10, 9999';
526:      for i:=1 to 16 do begin // reads each character of string in turn.
527:      //only reads to number 16 to avoid reading in the third comma, c.f.
    BalanceFileParser
528:        if Str[i]=',' then begin if flag=false then begin mk1:=i; flag:=true
529:                                                      end
530:        else begin
531:             mk2:=i; flag:=false; //comma positions located in mk1 and mk2
532:             end;           end;
533:                              end;
534:             //finds positions of the first two commas in line (only)
535:        CommaGap:=mk2-mk1; //calculates the distance between the two commas.
536: if CommaGap=9 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]);
537: if CommaGap=10 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]);
538: if TimeStr<>'' then Time:=strToInt(TimeStr); //this is the time in seconds.
539:
540: //*****************************************************************************
541: 1:
542:   with TimesIni do begin //TimesIni is the .ini file containing the current
    period settings
543:      Start:=ReadInteger('SecondGaps',
    'Interval'+Inttostr(intervalCounter)+'Start', -1);
544:      Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
    -1);
545:      //Start and Endd are variables containing start and end of interval in
    seconds.
546:                     end;
547: If Time>=Start then if Time<=Endd then begin
548:      TotalPerchMovements:=TotalPerchMovements+1;
549:      PerchFlag:=true; //signifies that there was a perch movement during current
    period
550:                                        end else
551:     begin
552:       if PerchFlag=false then begin//if no movements occured then do this
553:         Append(Results);
554:         WriteLn(Results, 0); //writes a zero in as result
555:         CloseFile(Results);
556:         IntervalCounter:=IntervalCounter+1;//ups IntervalCounter by one, next
    period
557:         goto 1;  //sends execution back to see if present perch movement fits
    within next period
558:                                end          else
559:           begin  //this code is executed if there are PerchMovements within
    current period
560:             Append(Results);
561:             WriteLn(Results, TotalPerchMovements);//log total perch movements
    within period
562:             CloseFile(Results);
563:             TotalPerchMovements:=0;   //zeroes this variable ready for next
    count
564:             IntervalCounter:=IntervalCounter+1; //ups Interval Counter by one,
```

```
    to next period
565:            PerchFlag:=False; //resets this flag
566:            goto 1;  //sends execution back to label 1 again
567:          end;
568:     end;
569: //*****************************************************************************
570: end; //end of 'for k:=1 to total do' routine.
571:
572: IntervalCounter:=IntervalCounter+1;//having read to end of file, this section
     fills in blank values so that output files have a uniform number of lines
573:   if PerchFlag=true then begin//if flag is true then there is remaining data to
     be written
574:      Append(Results);
575:      WriteLn(Results, TotalPerchMovements);
576:      CloseFile(Results);
577:      PerchFlag:=False;
578:
579: CloseFile(F);//closes current data file as it has been read to the end now...
580:      while start<>-1 do begin //while there are still periods remaining
581:      with TimesIni do
582:      Start:=ReadInteger('SecondGaps',
     'Interval'+intToStr(IntervalCounter)+'Start', -1);
583:      IntervalCounter:=IntervalCounter+1;
584:      Append(Results);
585:      if start<>-1 then WriteLn(Results, 0);//continues to add blank lines to
     ensure uniformity
586:      CloseFile(results);
587:                        end;
588:
589: PerchFlag:=false;//ensures this flag is set to false at this stage
590: intervalCounter:=0;
591:   for i:=1 to 80 do begin //this code creates the file containing current
     period start times
592:   //it runs the code 80 times, this is arbitrary and exceeds maximum no. of
     possible lines
593:   with TimesIni do
594:   Start:=ReadInteger('SecondGaps',
     'Interval'+inttostr(IntervalCounter)+'Start', -1);
595:   if start<>-1 then WriteLn(Times, (Inttostr(Start div 60)));//divides by 60
     hence minutes
596:   IntervalCounter:=IntervalCounter+1;
597:                     end;
598:   mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
     IntervalCheck:=False;
599:   time:=0;  //sets relevant variables back to default values ready for read of
     next file
600:
601:   CloseFile(Times);
602:   Total:=0; IntervalCounter:=0; TotalPerchMovements:=0; Start:=0; Endd:=0;
     //more resetting
603:        FileListBox1.color:=clwindow; //briefly alters color of FileListBox to
     denote end of read
604:
605:       end;//of 'if FileExist routine'**********************************
606:
607:
608: End;
609: if CurrentDate=edit8.text then goto 3;//if CurrentDate is finish date then this
     code stops
610: goto 2; //else it goes back to label 2 to begin reading next file in the
     sequence
611: 3:
612:
613: end;
614:
615:
616:
617:
618: procedure TForm1.RateOfMassGainButtonClick(Sender: TObject);
619:  begin
620: edit5.text:='data to c:\MassGainRate, MGR Prefix';
621: AssignFile(Results, 'c:\MassGainRate\MGRfeck '+FileEdit1.text);
622: AssignFile(F, label1.caption+'\'+FileEdit1.text);
623:   reset(F);
```

```
624:    rewrite(results);
625:     //this section of code pre-reads the file and calculates the number of
       lines.
626:    readLn(F, Str);
627:    while Str<>'' do begin Total:=Total+1;
628:       //EditTotal.text:=inttostr(total);
629:      //EditTotal.refresh;
630:    ReadLn(F, Str);
631:                       end;
632: CloseFile(F); //end of line counter- no. of lines held in Total.
633:    Edit3.text:=inttostr(total);
634:    Reset(F);
635:
636:    for k:=1 to total do begin //reads through each line of the textfile, this is
       the ROUTINE
637:       ReadLn(F, Str); //loads current line of textfile into Str variable.
638: PreviousMass:=Mass;
639: PreviousTime:=Time;
640:
641: StrLength:=Length(str);  //calculate length of string
642:     for i:=1 to StrLength do begin  //read each character of Str and find
       commas
643:     if Str[i]=',' then begin if flag=false then begin mk1:=i; flag:=true end
       else begin
644:     mk2:=i; flag:=false; end; end; end; //gives the positions of the two commas
645:     commagap:=mk2-mk1;
646:     d1:=mk1+2; d2:=mk1+3; d3:=mk1+4; d4:=mk1+5; d5:=mk1+6;
647:  if IsTimeInSeconds(Str)=true then begin
648: if mk1=2 then  if str<>'' then timeStr:='0';
649:  if mk1=3 then  if str<>'' then timeStr:=concat(str[1]);
650:   if mk1=4 then if str<>'' then timeStr:=concat(str[1]+str[2]);
651:    if mk1=5 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
652:     if mk1=6 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
653:      if mk1=10 then close;
654:                                   end
655: else//if time is in milliseconds instead of seconds in the textfile
656:  begin
657:  if mk1<=4 then timeStr:='0';
658:   if mk1=5 then if str<>'' then timeStr:=str[1];
659:     if mk1=6 then if str<>'' then timeStr:=concat(str[1]+str[2]);
660:      if mk1=7 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]);
661:       if mk1=8 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
662:        if mk1=9 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
663:         if mk1=10 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]+str[6]);
664:  end; //of if milliseconds code routine
665: if Timestr<>'' then time:=strToInt(TimeStr);
666:
667: realTime:=concat(str[mk2+2]+str[mk2+3]+str[mk2+4]+str[mk2+5]+str[mk2+6]
668:                  +str[mk2+7]+str[mk2+8]+str[mk2+9]);
669:
670: if CommaGap=4 then if str<>'' then MassStr:=concat(str[d1]+str[d2]);//for two
       digit mass
671: if CommaGap=6 then if str<>'' then
       MassStr:=Concat(str[d1]+str[d2]+str[d3]+str[d4]);//with four
672:  if CommaGap=7 then if str<>'' then
       MassStr:=Concat(str[d1]+str[d2]+str[d3]+str[d4]+str[d5]);
673: if MassStr<>'' then mass:=strtoFloat(MassStr); //contains value of mass reading
       for current line
674: // now Mass and Time are held in variables of those names.
675: DeltaMass:=Mass-PreviousMass;
676: DeltaTime:=Time-PreviousTime;
677:  if DeltaTime=0 then DeltaTime:=1;
678: RateOfMassChange:=(DeltaMass / DeltaTime)*100000;
679: if initialise=false then WriteLn(Results, intToStr(time)+',
       '+floattostrF(RateOfMassChange, ffFixed, 6, 0)
680: +', '+realTime);
681: initialise:=false;
682:
683:
684:      end;
```

```
685:   CloseFile(F);
686:   CloseFile(Results);
687:   Total:=0;  DeltaMass:=0; Mass:=0;  PreviousMass:=0;  Deltatime:=0;
688:   PreviousTime:=0; RateOfMassChange:=0; Initialise:=true;
689: end;
690:
691:
692:
693: procedure TForm1.BtnBaldataClick(Sender: TObject);
694: var RealTime, originalFileName:string;
695: begin
696:  //this code is reading giant 'EveryBalanceOutput' files and parsing the data
697:  // by only allowing lines with over 20 repeats to be written to textfile
698: edit5.text:='data to c:\LRE2 (Mod Prefix)';
699: AssignFile(results, 'c:\LRE2\Mod'+FileEdit1.text);
700: Rewrite(Results);
701:    AssignFile(F, label1.caption+'\'+FileEdit1.text);
702:    reset(F);
703:      //this section of code pre-reads the file and calculates the number of
     lines
704:  readLn(F, Str);
705:  while Str<>'' do begin Total:=Total+1;
706:     //EditTotal.text:=inttostr(total);
707:     //EditTotal.refresh;
708:  ReadLn(F, Str); end;
709:   CloseFile(F); //end of line counter- variable is called Total.
710:   Edit3.text:=inttostr(total);
711:    Reset(F);
712: ////////////////////////////////////////////////////////////////////////////
713: for k:=1 to total-1 do begin
714:  //reads through each line of the textfile-ROUTINE***************
715: PreviousMassStr:=MassStr;
716: PreviousTime:=Time;
717: readLn(F, Str); //reads line into Str variable
718: StrLength:=Length(Str); //reads string length into 'Length' variable
719: for i:=1 to StrLength do begin//finds commas, four of them
720: if str[i]=',' then begin
721:    if commaCount=1 then begin mk1:=i; CommaCount:=2 end else
722:     if CommaCount=2 then begin mk2:=i; CommaCount:=3 end else
723:      if CommaCount=3 then begin mk3:=i; CommaCount:=4 end else
724:       if CommaCount=4 then begin mk4:=i; CommaCount:=1;end; //this puts the
     comma positions into mk1,2,3 & 4
725:                     end;    //of comma finding routine
726:   CommaGapA:=mk2-mk1;
727:    CommagapB:=mk3-mk2;
728:     CommaGapC:=mk4-mk3;
729:
730: if CommaGapA=6 then
     MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]+str[mk1+5]);
731:  if str[mk1+3]=' ' then if CommaGapA=7 then
732:   MassStr:=concat(str[mk1+2]+str[mk1+4]+str[mk1+5]+str[mk1+6]);
733:   if CommaGapA=7 then
     MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]+str[mk1+5]+str[mk1+6]);
734:   //last four lines create string Variable containing mass reading, including
     sign if present
735: if mk1=5 then  if str<>'' then timeStr:='0';
736:  if mk1=6 then  if str<>'' then timeStr:=concat(str[1]);
737:   if mk1=7 then if str<>'' then timeStr:=concat(str[1]+str[2]);
738:    if mk1=8 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
739:     if mk1=9 then if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
740:      if mk1=10 then close;
741:  if TimeStr<>'' then IntTime:=strtoint(timeStr);//this is the time in seconds
     as an integer
742:
     realTime:=Concat(str[mk2+2]+str[mk2+3]+str[mk2+4]+str[mk2+5]+str[mk2+6]+str[
    mk2+7]+
743:                  str[mk2+8]+str[mk2+9]);
744:
745:   PositionBird:=str[mk4+2];//gives either a ?, F or D to denote position
746:
747: if CommaGapC=3 then Repeats:=str[mk3+2];
748:  if CommaGapC=4 then Repeats:=Concat(str[mk3+2]+str[mk3+3]);
749:   if CommaGapC=5 then Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]);
```

```
750:     if CommaGapC=6 then
      Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]);
751:       if CommaGapC=7 then
      Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]+str[mk3+6]);
752:        if CommaGapc=8 then end;
753: //From here Mass, Time, PreviousMass, PreviousTime and
754: // realTime and Position are all available for manipulations
755:
756: RepeatsInt:=StrToInt(Repeats); //only writes lines with over 20 repeats to new
      filename
757: if RepeatsInt>20 then begin
758:    Append(Results);
759:    WriteLn(Results, Str);
760:     CloseFile(Results);
761:
762:
763:                         end;
764:                         end;
765:  messagebeep(MB_ICONEXCLAMATION);  Total:=0;
766:     Reset(F);     CloseFile(F);
767:  end;
768:
769:
770:
771:
772:
773: //this procedure should by now be largely redundant as all the files needing
774: // attention are now sorted out
775: procedure TForm1.BtnCorrectErrantMassFilesClick(Sender: TObject);
776: var RealTime, originalFileName:string;
777: begin
778:  //this code is reading giant 'EveryBalanceOutput' files and shifting the mass
      values down
779:  //one place, such that 3 goes to 4, 4 to 5 etc...
780:  edit5.text:='Data in c:\LRE2a';
781:  Commacount:=1;
782: AssignFile(results, 'c:\LRE2a\'+FileEdit1.text);
783: Rewrite(Results);
784:     AssignFile(F, label1.caption+'\'+FileEdit1.text);
785:     reset(F);
786:       //this section of code pre-reads the file and calculates the number of
      lines
787:  readLn(F, Str);
788:  while Str<>'' do begin Total:=Total+1;
789:     //EditTotal.text:=inttostr(total);
790:     //EditTotal.refresh;
791:  ReadLn(F, Str); end;
792:   CloseFile(F); //end of line counter- variable is called Total.
793:   Edit3.text:=inttostr(total);
794:    Reset(F);
795: ////////////////////////////////////////////////////////////////////////////
796: for k:=1 to total-1 do begin
797:  //reads through each line of the textfile-ROUTINE***************
798: PreviousMassStr:=MassStr;
799: PreviousTime:=Time;
800: readLn(F, Str); //reads line into Str variable
801: StrLength:=Length(Str); //reads string length into 'Length' variable
802: for i:=1 to StrLength do begin//finds commas, four of them
803: if str[i]=',' then begin
804:    if commaCount=1 then begin mk1:=i; CommaCount:=2 end else
805:     if CommaCount=2 then begin mk2:=i; CommaCount:=3 end else
806:      if CommaCount=3 then begin mk3:=i; CommaCount:=4 end else
807:       if CommaCount=4 then begin mk4:=i; CommaCount:=1;end; //this puts the
      comma positions into mk1,2,3 & 4
808:                       end;    //of comma finding routine
809:   CommaGapA:=mk2-mk1;
810:    CommagapB:=mk3-mk2;
811:     CommaGapC:=mk4-mk3;
812:
813: if CommaGapA=6 then
      MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]+str[mk1+5]);
814:  if str[mk1+3]=' ' then if CommaGapA=7 then
815:   MassStr:=concat(str[mk1+2]+str[mk1+4]+str[mk1+5]+str[mk1+6]);
816:   if CommaGapA=7 then
```

```
     MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]+str[mk1+5]+str[mk1+6]);
817:   //last four lines create string Variable containing mass reading, including
     sign if present
818: if mk1=5 then  if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
819:  if mk1=6 then  if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
820:   if mk1=7 then if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]+str[6]);
821:    if mk1=8 then if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]+str[6]+str[7]);
822:     if mk1=9 then if str<>'' then
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]+str[6]+str[7]+str[8]);
823:      if mk1=10 then close;
824: if TimeStr<>'' then IntTime:=strtoint(timeStr);//this is the time in seconds as
     an integer
825:
     realTime:=Concat(str[mk2+2]+str[mk2+3]+str[mk2+4]+str[mk2+5]+str[mk2+6]+str[
    mk2+7]+
826:                  str[mk2+8]+str[mk2+9]);
827:
828:   PositionBird:=str[mk4+2];//gives either a ?, F or D to denote position
829:
830: if CommaGapC=3 then Repeats:=str[mk3+2];
831:  if CommaGapC=4 then Repeats:=Concat(str[mk3+2]+str[mk3+3]);
832:   if CommaGapC=5 then Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]);
833:    if CommaGapC=6 then
     Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]);
834:     if CommaGapC=7 then
     Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]+str[mk3+6]);
835:      if CommaGapc=8 then end;
836: //From here Mass, Time, PreviousMass, PreviousTime and
837: // realTime and Position are all available for manipulations
838:
839: if PreviousMassStr<>'' then begin
840: Append(results);
841: if strToInt(repeats)>20 then
842: writeLn(Results,timestr+', '+PreviousmassStr+', '+Realtime+', '+Repeats+',
     '+PositionBird);
843: CloseFile(Results);
844:                              end;
845:
846:   end;
847:  messagebeep(MB_ICONEXCLAMATION);
848:  Total:=0; CommaCount:=1; mk1:=0; mk2:=0; mk3:=0; mk4:=0;
849: end;
850:
851:
852:
853: procedure TForm1.BtnCorrectDriftClick(Sender: TObject);
854: begin
855: BtnCorrectDrift.default:=true;
856: BtnCorrectDrift.enabled:=false;
857:  //this code is reading mass files and separating drift from actual mass
     readings.
858: edit5.text:='data to c:\MassData, DC prefix';
859: AssignFile(results, 'c:\MassData\DC'+FileEdit1.text);
860: Rewrite(Results);
861: CloseFile(Results);
862:    AssignFile(F, label1.caption+'\'+FileEdit1.text);
863:    reset(F);
864:      //this section of code pre-reads the file and calculates the number of
     lines
865:  readLn(F, Str);
866:  while Str<>'' do begin Total:=Total+1;
867:    //EditTotal.text:=inttostr(total);
868:     //EditTotal.refresh;
869:  ReadLn(F, Str); end;
870:   CloseFile(F); //end of line counter- variable is called Total.
871:   Edit3.text:=inttostr(total);
872:    Reset(F);
873: ////////////////////////////////////////////////////////////////////////////
874: for k:=1 to total{-1} do begin
875:  PreviousTime:=IntTime;
876:    PreviousMass:=Mass;
```

```
877:  //reads through each line of the textfile-ROUTINE***************
878:
879: readLn(F, Str); //reads line into Str variable
880: StrLength:=Length(Str); //reads string length into 'Length' variable
881:
882:
883: for i:=1 to StrLength do begin//finds commas, four of them
884: if str[i]=',' then begin
885:    if commaCount=1 then begin mk1:=i; CommaCount:=2 end else
886:     if CommaCount=2 then begin mk2:=i; CommaCount:=3 end else
887:      if CommaCount=3 then begin mk3:=i; CommaCount:=4 end else
888:       if CommaCount=4 then begin mk4:=i; CommaCount:=1;end; //this puts the
     comma positions into mk1,2,3 & 4
889:                     end;  end;  //of comma finding routine
890:   CommaGapA:=mk2-mk1;
891:    CommagapB:=mk3-mk2;
892:     CommaGapC:=mk4-mk3;
893:
894: if commaGapA=7 then if str[mk1+3]=' ' then
895:  MassStr:=concat(str[mk1+4]+str[mk1+5]+str[mk1+6]) else
896:    MassStr:=concat(str[mk1+3]+str[mk1+4]+str[mk1+5]+str[mk1+6]);
897:      Mass:=strtofloat(MassStr);
898:        if str[mk1+2]='-' then Mass:=Mass*-1;
899:           if CommaGapA<>7 then close; //stops program if digits are not in
     standard format
900:
901:
902: if mk1=5 then  if str<>'' then timeStr:='0';
903:  if mk1=6 then  if str<>'' then timeStr:=concat(str[1]);
904:   if mk1=7 then if str<>'' then timeStr:=concat(str[1]+str[2]);
905:    if mk1=8 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
906:     if mk1=9 then if str<>'' then                                          .
     timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
907:      if mk1=10 then close;
908:  if TimeStr<>'' then IntTime:=strtoint(timeStr);//this is the time in seconds
     as an integer
909:
     realTime:=Concat(str[mk2+2]+str[mk2+3]+str[mk2+4]+str[mk2+5]+str[mk2+6]+str[
    mk2+7]+
910:                 str[mk2+8]+str[mk2+9]);
911:
912:   PositionBird:=str[mk4+2];//gives either a ?, F or D to denote position
913:
914: if CommaGapC=3 then Repeats:=str[mk3+2];
915:  if CommaGapC=4 then Repeats:=Concat(str[mk3+2]+str[mk3+3]);
916:   if CommaGapC=5 then Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]);
917:    if CommaGapC=6 then
     Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]);
918:     if CommaGapC=7 then
     Repeats:=Concat(str[mk3+2]+str[mk3+3]+str[mk3+4]+str[mk3+5]+str[mk3+6]);
919:      if CommaGapc=8 then Repeats:='10000';
920:       if repeats<>'' then RepeatsExt:=strtofloat(repeats);
921: //From here Mass, PreviousMass, IntTime, and
922: // realTime, RepeatsExt and PositionBird are all available for manipulations
923:
924: PreviousBirdMass:=BirdMass;
925:  Difference:=mass-PreviousMass;
926:   if Difference>=10 then begin
927:                          TareValue:=PreviousMass;
928:                          BirdMass:=Mass-TareValue;
929:                          NewBirdMassFlag:=True;
930:                          end else
931:    if Difference<=-10 then begin
932:                            TareValue:=Mass;
933:                            BirdMass:=PreviousMass-TareValue;
934:                            NewBirdMassFlag:=True;
935:                            end else
936:     if Abs(Difference)<5 then if abs(Difference)>0 then if Mass-TareValue>=10
     then
937:          begin
938:          BirdMass:=Mass-TareValue;
939:           if RepeatsExt>20 then NewBirdMassFlag:=True else
     NewBirdMassFlag:=False;/////////
940:          end else
```

```
941:            NewBirdMassFlag:=False;
942:
943:   if NewBirdMassFlag=True then
944:    if (IntTime-PreviousTime)>0 then
945:     if (abs(BirdMass-PreviousBirdMass)/(IntTime-PreviousTime))<0.1 then
946:    begin
947:    Append(results);
948:    WriteLn(Results, InttoStr(IntTime)+', '+FloattoStr(BirdMass)+', '+RealTime);
949:    CloseFile(Results);
950:    end;
951:
952: end;
953:
954:
955:  messagebeep(MB_ICONEXCLAMATION); btncorrectdrift.enabled:=true;
956:   Reset(F);    CloseFile(F);
957:    Total:=0; CommaCount:=1; mk1:=0; mk2:=0; mk3:=0; mk4:=0;CommaGapA:=0;
      CommaGapB:=0; CommaGapC:=0;
958: end;
959:
960:
961:
962:
963:
964:
965: procedure TForm1.ButtonCombineFilesClick(Sender: TObject);
966: var TemporaryLocale, FileBeingAdded, AccumulatedResults: textfile;
967:  StrNew, StrCum :String;
968: begin
969: messageBeep(0);
970: btnTransformFile.Default:=false;
971: ButtonCombineFiles.default:=true;//makes this the default control
972: edit5.text:='New file in c:\Temp'; //indicates location of results file
973: FileListBox1.color:=claqua; FileListBox1.refresh;//flashes the list box
974: Assignfile(TemporaryLocale, 'c:\Temp\temp.txt');//sets up temporary file
      location
975: AssignFile(FileBeingAdded, label1.caption+'\'+FileEdit1.text);//name of file
      being added
976: AssignFile(AccumulatedResults, 'c:\Temp\'+Edit4.text+'.txt'); //name of
      Accumulated file
977: if FileExists('c:\Temp\'+Edit4.text+'.txt') =false then
978:  rewrite(AccumulatedResults)   ; reset(AccumulatedResults);//if file is new
      then it is created and set ready for writing
979:  if FileExists('c:\Temp\temp') =false then //if file exists already then the
      temporary location is reset instead
980:   rewrite(TemporaryLocale)   ; reset(TemporaryLocale);
981:    if FileExists(label1.caption+'\'+FileEdit1.text) =false then
982:     rewrite(FileBeingAdded)   ; reset(FileBeingAdded);
983: //all files are presently open for reading and writing
984: strNew:='anythingbutnothing';//sets this string to an arbitrary non-zero value
985: While StrNew<>'' do begin//while still reading from the FileBeingAdded this
      code continues to fire
986: ReadLn(AccumulatedResults, StrCum);//read first line of Accumulated file into
      memory
987: ReadLn(FileBeingAdded, StrNew);//read first line of AddedFile into memory
988: Append(TemporaryLocale); if StrNew<>'' then//writes accumulated string, plus
      line from FileBeingAdded, into the temp. location
989: if StrCum<>'' then WriteLn(TemporaryLocale, StrCum+'; '+strNew)
990:  else{if no previous Cumulative data} WriteLn(TemporaryLocale, strNew);
991: CloseFile(TemporaryLocale);
992: end;
993:
994: CloseFile(AccumulatedResults);
995: Rewrite(AccumulatedResults);
996: Reset(TemporaryLocale);
997: StrNew:='AnythingButNothing';//again sets this string to an arbitrary non-zero
      value
998:  While StrNew<>'' do begin
999:  ReadLn(TemporaryLocale, StrNew);//reads first line from Temporary fie
1000:   Append(AccumulatedResults); //opens results file for appending
1001:    if StrNew<>'' then WriteLn(AccumulatedResults, StrNew);//writes each line
      from Temp to results file
1002:     CloseFile(AccumulatedResults);
1003:      end;
```

```
1004:       rewrite(TemporaryLocale);//clears temporary file for next cycle
1005:       CloseFile(TemporaryLocale);
1006:        CloseFile(FileBeingAdded);
1007:         FileListBox1.color:=clwindow; fileListBox1.refresh;
1008: end;
1009:
1010:
1011: procedure TForm1.ButtonAwardPerchScoreClick(Sender: TObject);
1012: label 1; label 2; label 3;
1013: var CurrentDate, PreviousDuration : String; totalhopGap, totaliser:extended;
1014:
1015: begin
1016: 2:
1017: duration:='0';
1018:   totaliser:=0;
1019: CurrentDate:=edit6.text;//sets starting date using default value
1020:
1021:
1022:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1023:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
       back to 1 again
1024:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1025:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
       ['+edit6.text+']';
1026:   edit7.refresh;  //displays the file (with filepath) currently being parsed
       (if it exists)
1027: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
1028: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt')
1029: //checks whether file exists or not, if so then the code below is executed
1030:  then begin
1031: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt');
1032:
1033: edit5.text:='Data to c:\ParsedPerchData, with Alt. prefix';//indicates path to
       results
1034: intervalCounter:=0; DisplayValue:=0; TotalDisplayValue:=0;  //resets some
       variables
1035:  ButtonAwardPerchScore.default:=true;//makes this the default control
1036: AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
1037: AssignFile(Results, 'c:\ParsedPerchData\Alt1.Perch '+IntToStr(CageNumber)+'
       ['+CurrentDate+'].txt');;
1038: // AssignFile(F, label1.caption+'\'+FileEdit1.text);
1039:   reset(F); //resets source file ready for reading
1040:   rewrite(results);//clears results file reading for writing
1041:   Rewrite(times);
1042:   //this section of code pre-reads the file and calculates the number of lines.
1043:   readLn(F, Str);
1044:    FileListBox1.color:=cllime;filelistbox1.refresh;//flashes list box to denote
       finish
1045:   while Str<>'' do begin Total:=Total+1;
1046:    //EditTotal.text:=inttostr(total);
1047:     //EditTotal.refresh;
1048:   ReadLn(F, Str);
1049:                     end;
1050: CloseFile(F); //end of line counter- no. of lines held in Total.
1051:   Edit3.text:=inttostr(total);//displays total no. of lines
1052:   Reset(F); //resets data textfile for next read of values
1053: //*****************************************************************************
1054:   for k:=1 to total do begin //reads through each line of the textfile, this is
       the ROUTINE
1055:       ReadLn(F, Str); //loads current line of textfile into Str variable.
1056:       CommaFlag:=1;//sets default value
1057:
1058:       for i:=1 to Length(str) do begin// reads each character of whole string in
       turn.
1059: if str[i]=',' then begin
1060:    if Commaflag=1 then begin mk1:=i; Commaflag:=2 end else
1061:     if Commaflag=2 then begin mk2:=i; Commaflag:=3 end else
1062:      if Commaflag=3 then begin mk3:=i; Commaflag:=1 end;
1063:                     end;  end; //above routine finds the positions of the commas
1064:
1065:
1066:        CommaGap:=mk2-mk1; //calculates the distance between the two commas.
```

```
1067: if CommaGap=9 then
       timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]+str[9]+str[10]);
1068: if CommaGap=10 then
       timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]+str[9]+str[10]+str[11]);
1069: PreviousTime:=TimeA;//keeps a running record of previous time value
1070: if TimeStr<>'' then TimeA:=strToInt(TimeStr); //this is the time in
       milliseconds.
1071: HopGap:=TimeA-PreviousTime; //Time in milliseconds between present and previous
       hops
1072: if HopGap<=0 then HopGap:=1;//prevents divide by zero errors, 1ms is a
       negligible value
1073: PreviousPerchID:=PerchID;//keeps a running note of which perch was active
1074: PerchID:=str[1]; //Identity of perch which is depressed
1075: PreviousDuration:=Duration;//keeps note of previous duration value
1076: duration:=''; //clears durations string prior to writing a new value into it
1077: for j:=mk3+2 to Length(Str) do
1078:  duration:=duration+Str[j];//gives string 'Duration' with duration in Msecs
1079: Time:=abs(TimeA div 1000); //converts time into seconds in readiness for next
       section of code
1080: //*****************************************************************************
1081: 1:
1082:   with TimesIni do begin//refers to .ini file for start and finish times
1083:     Start:=ReadInteger('SecondGaps',
       'Interval'+Inttostr(intervalCounter)+'Start', -1);
1084:     Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
       -1);
1085:                    end;
1086: If Time>=Start then if Time<=Endd then begin
1087: //the following block of code correctly sets the NewPerchScore for each cage
1088: //*****************************************************************************
1089: if CageNumber<=2 then begin //for cages 1 and 2 only
1090:    if PreviousPerchID=PerchID                then NewPerchScore:=1;
1091:    if PreviousPerchID<>PerchID               then NewPerchScore:=1.2;
1092:    if PreviousPerchID='1' then if PerchID='2' then NewPerchScore:=1.4;
1093:    if PreviousPerchID='2' then if PerchID='1' then NewPerchScore:=1.4;
1094:    if PreviousPerchID='3' then if PerchID='4' then NewPerchScore:=1.4;
1095:    if PreviousPerchID='4' then if PerchID='3' then NewPerchScore:=1.4;
1096:                    end;
1097:  if CageNumber=3 then begin //for cage 2 only
1098:    if PreviousPerchID=PerchID                then NewPerchScore:=1;
1099:    if PreviousPerchID<>PerchID               then NewPerchScore:=1.2;
1100:    if PreviousPerchID='1' then if PerchID='4' then NewPerchScore:=1.4;
1101:    if PreviousPerchID='4' then if PerchID='1' then NewPerchScore:=1.4;
1102:    if PreviousPerchID='2' then if PerchID='3' then NewPerchScore:=1.4;
1103:    if PreviousPerchID='3' then if PerchID='2' then NewPerchScore:=1.4;
1104:                    end;
1105:   if CageNumber>=4 then begin //for cages 4 to 8 only
1106:    if PreviousPerchID=PerchID                then NewPerchScore:=1;
1107:    if PreviousPerchID<>PerchID               then NewPerchScore:=1.2;
1108:    if PreviousPerchID='1' then if PerchID='3' then NewPerchScore:=1.4;
1109:    if PreviousPerchID='3' then if PerchID='1' then NewPerchScore:=1.4;
1110:    if PreviousPerchID='2' then if PerchID='4' then NewPerchScore:=1.4;
1111:    if PreviousPerchID='4' then if PerchID='2' then NewPerchScore:=1.4;
1112:                    end;
1113: //*****************************************************************************
1114:
1115: // remarked out on 11/05/99***if PerchID<>PreviousPerchID then
       NewPerchScore:=0.5 else NewPerchScore:=0;
1116:  if HopGap=0 then HopGap:=1;
1117:   if StrToInt(Duration)<=0 then Duration:='12961';
1118:    if StrToInt(PreviousDuration)<=0 then PreviousDuration:='12961';
1119: // remarked out on 11/05/99***DisplayValue:=(2500 /
       StrToInt(Duration))+(NewPerchScore+(300 / HopGap));
1120:  //totalhopgap:=totalhopgap+hopgap;
1121:  //totaliser:=totaliser+1;
1122:  //edit9.text:=floattostr(hopgap); edit9.refresh;
1123:  //edit10.text:=FloattoStr(NewPerchScore);  edit10.refresh;
1124:  DisplayValue:=StrToInt(Duration);//*NewPerchScore;
1125:    TotalDisplayValue:= TotalDisplayValue+DisplayValue;
1126:  //above code calculates the display index and totals it for the interval in
       hand
1127:     PerchFlag:=true;
1128:                                        end else
1129:     begin
```

```
1130:       if PerchFlag=false then begin
1131:         Append(Results);
1132:         WriteLn(Results, 0);//if no data for that period then a zero is awarded
1133:         CloseFile(Results);
1134:         IntervalCounter:=IntervalCounter+1;//clicks forward to the next
      interval
1135:         goto 1;
1136:                                end          else
1137:           begin
1138:            Append(Results);
1139:            WriteLn(Results, floattostrF(TotalDisplayValue, ffFixed, 6, 5));
1140:            CloseFile(Results); //writes the perch display index into the
      results textfile
1141:            TotalDisplayValue:=0; //resets this variable
1142:            IntervalCounter:=IntervalCounter+1; //clicks forward onto next
      interval
1143:            PerchFlag:=False;
1144:            goto 1; //sends execution point back to 1
1145:          end;
1146:     end;
1147: //*****************************************************************************
1148: end; //end of 'for k:=1 to total do' routine.
1149:
1150: IntervalCounter:=IntervalCounter+1;//clicks forward to next interval
1151:   if PerchFlag=true then begin
1152:      Append(Results);
1153:      WriteLn(Results, floattostrF(TotalDisplayValue, ffFixed, 7, 6));
1154:      CloseFile(Results);//writes remaining data to results, if it is present
1155:      PerchFlag:=False;
1156:
1157: CloseFile(F);
1158:      while start<>-1 do begin
1159:      with TimesIni do
1160:      Start:=ReadInteger('SecondGaps',
      'Interval'+intToStr(IntervalCounter)+'Start', -1);
1161:      IntervalCounter:=IntervalCounter+1;
1162:      Append(Results);
1163:      if start<>-1 then WriteLn(Results, 0);//fills in zero values if there is
      no more data
1164:      CloseFile(results);
1165:                          end;
1166:
1167: PerchFlag:=false;
1168: intervalCounter:=0;
1169:
1170:    for i:=1 to 80 do begin//this code creates the current time divisions file
1171:    with TimesIni do
1172:    Start:=ReadInteger('SecondGaps',
      'Interval'+inttostr(IntervalCounter)+'Start', -1);
1173:      hours:=start div 3600;
1174:     minutes:=(start-(hours*3600)) div 60;
1175:      hourStr:=intToStr(Hours); if Length(hourstr)=1 then hourStr:='0'+HourStr;
1176:       MinStr:=IntToStr(Minutes); if Length(MinStr)=1 then MinStr:='0'+Minstr;
1177:     TimeStr:=Concat(HourStr+':'+MinStr);
1178:    if start<>-1 then WriteLn(Times, (Inttostr(Start div 60))+', '+TimeStr);
1179:    IntervalCounter:=IntervalCounter+1;
1180:                       end;
1181:    mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
      IntervalCheck:=False;
1182:    timeA:=0;
1183:
1184:    CloseFile(Times); //closes and tidies up
1185:    Total:=0; IntervalCounter:=0; TotalDisplayValue:=0; Start:=0; Endd:=0;
1186:          FileListBox1.color:=clwindow;
1187:       //   edit10.text:=floattostr(totalhopgap/totaliser);
1188:  end;
1189:    end;
1190: if CurrentDate=edit8.text then goto 3;
1191:  goto 2;
1192:    3:
1193: end; //of AwardPerchScore routine
1194:
1195:
1196: procedure TForm1.BtnMassGainPerIntClick(Sender: TObject);
```

```
1197: var AvoidRepeatsCheck: integer;
1198: i, mk1, Starttt:integer; TimeStr, str:string;
1199: CurrentDate:string;
1200: label 1; label 2; label 3;
1201: begin
1202: //code highlighted below is responsible for reading files sequentially
1203: ////////////////////////////////////////////////////////////////////////////
1204: CurrentDate:=edit6.text;//sets current date
1205:
1206: 2:
1207:  FileListBox1.color:=clwhite;filelistbox1.refresh;//flashes list box to denote
      cycling
1208:  CageNumber:=CageNumber+1; if CageNumber=9 then //flips to next day and cage 1
1209:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1210:  edit7.text:=label1.caption+'\'+'Cage '+IntToStr(CageNumber)+'
      ['+edit6.text+']';
1211:   edit7.refresh;
1212: edit6.text:=CurrentDate; edit6.refresh; //displays current date and updates
      display
1213: if FileExists(label1.caption+'\'+'Cage'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt')
1214:  then begin //if File is present in selected folder then run the following code
1215:
1216: assignFile(F,  label1.caption+'\'+'Cage'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt');
1217: ////////////////////////////////////////////////////////////////////////////
1218:  FileListBox1.color:=cllime;filelistbox1.refresh;//flashes the list box to
      denote end of cycle
1219: btnTransformFile.Default:=true;//makes the button the default control
1220: starttt:=0;
1221: AssignFile(Results, 'c:\temp\MGR'+IntToStr(CageNumber)+'
      ['+CurrentDate+'].txt');
1222:    reset(F);//resets the current data file
1223:       rewrite(results); //clears current results file ready for writing
1224:
1225:
1226: /////////
1227: IntervalTimeCheck:=0;//starts at zero hours and minutes
1228:   BtnMassGainPerInt.default:=true;
1229:     BtnMassGainPerInt.enabled:=false;
1230: //this section of code pre-reads the file and calculates the number of lines
1231:  readLn(F, Str);
1232:  while Str<>'' do begin Total:=Total+1;
1233:    //EditTotal.text:=inttostr(total);
1234:     //EditTotal.refresh;
1235:  ReadLn(F, Str); end;
1236:   CloseFile(F); //end of line counter- variable is called Total.
1237:   Edit3.text:=inttostr(total);
1238:    Reset(F);
1239: ////////////////////////////////////////////////////////////////////////////
1240: for k:=1 to total do begin  //start of main routine**********************
1241:  TimeFromPreviousLine:=TimeFromCurrentLine;//keeps rolling record of time
1242:   PreviousMass:=Mass;                      //keeps rolling record of mass
1243:   PreviousMassStr:=MassStr;                //keeps rolling record of mass as
      string
1244:    PreviousRealTime:=RealTime;             //keeps rolling record of time as
      string
1245:     StartingMass:=EndingMass;              //keeps rolling record of start mass
1246:      StartingTime:=EndingTime;             //keeps rolling record of start time
1247:  readLn(F, Str); //reads line into Str variable
1248: StrLength:=Length(Str); //reads string length into 'Length' variable
1249:
1250: for i:=1 to StrLength do begin//finds commas
1251: if str[i]=',' then begin
1252:    if commaCount=1 then begin mk1:=i; CommaCount:=2 end else
1253:     if CommaCount=2 then begin mk2:=i; CommaCount:=1 end //this puts the comma
      positions into mk1,2
1254:                   end;  end;  //of comma finding routine
1255:   CommaGapA:=mk2-mk1;
1256:  if commaGapA=5 then  MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]);
1257:   if commaGapA=4 then  MassStr:=concat(str[mk1+2]+str[mk1+3]);
1258:    if commaGapA=6 then
      MassStr:=concat(str[mk1+2]+str[mk1+3]+str[mk1+4]+str[mk1+5]);
1259:     Mass:=strtofloat(MassStr); //current mass is now extracted into memory
```

```
1260:
1261:  if IsTimeInSeconds(Str)=true then begin //makes function call to determine
       time format, which varies between early and late files
1262: if mk1=2 then  if str<>'' then timeStr:='0';
1263:  if mk1=3 then  if str<>'' then timeStr:=concat(str[1]);
1264:   if mk1=4 then if str<>'' then timeStr:=concat(str[1]+str[2]);
1265:    if mk1=5 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
1266:     if mk1=6 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
1267:      if mk1=10 then close;
1268:                                    end
1269: else//if time is in milliseconds instead of seconds in the textfile
1270:  begin
1271:  if mk1<=4 then timeStr:='0';
1272:   if mk1=5 then if str<>'' then timeStr:=str[1];
1273:    if mk1=6 then if str<>'' then timeStr:=concat(str[1]+str[2]);
1274:     if mk1=7 then if str<>'' then timeStr:=concat(str[1]+str[2]+str[3]);
1275:      if mk1=8 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]);
1276:       if mk1=9 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]);
1277:        if mk1=10 then if str<>'' then
       timeStr:=concat(str[1]+str[2]+str[3]+str[4]+str[5]+str[6]);
1278:  end; //of if milliseconds code routine
1279:
1280:  if TimeStr<>'' then TimeFromCurrentLine:=strtoint(timeStr);//this is the time
       in seconds as an integer
1281:
       realTime:=Concat(str[mk2+2]+str[mk2+3]+str[mk2+4]+str[mk2+5]+str[mk2+6]+str[
      mk2+7]+
1282:                  str[mk2+8]+str[mk2+9]);
1283: CurrentTimeInterval:=IntervalTimeCheck;  {add 1800 per 30 minutes}
1284: DifferenceA:=TimeFromCurrentLine-CurrentTimeInterval; //TimeFromCurrentLine is
       simply Integer Time, current
1285:  if DifferenceA>=0 then begin//if difference is positive then time from file
       exceeds IntervalTimeCheck
1286:   DifferenceB:=CurrentTimeInterval-TimeFromPreviousLine;//difference between
       previous+IntervalTimeCheck
1287: //if DifferenceB>0 then if MessageDlg('DifferenceB is greater than zero,
       Exit?',
1288: // mtInformation, [mbYes, mbNo], 0) = mrYes then close;
1289:
1290:  //above code establishes whether time before or after Current Interval is
       closest
1291:    ////////////////////////////
1292:     Minutes:=IntervalTimeCheck div 60; while minutes>=60 do
       minutes:=minutes-60;
1293:      MinuteStr:=IntToStr(Minutes); if length(minuteStr)=1 then
       MinuteStr:=concat('0'+MinuteStr);
1294:      hourStr:=inttostr(abs(IntervalTimeCheck div 3600)); if Length(hourStr)=1
       then HourStr:=concat('0'+hourStr);
1295:       //if the hour is only one digit long then a zero is appended at the front
       of it
1296:       TimeStr:=concat(HourStr+':'+MinuteStr); //makes up the time string by
       concatenating the two
1297:       ////////////////////////////////////this code converts 'IntervalTimeCheck'
       to Hours:Minutes
1298:    if IntervalTimeCheck>86399 then Finished:=true; //finishes execution once
       end of day is reached
1299:     IntervalTimeCheck:=IntervalTimeCheck+1800; //clicks IntervalTimeCheck
       variable forward by 30 minutes
1300:           Append(results);
1301:
1302: if DifferenceB<=DifferenceA then //if so use PreviousMass or else use MassStr
1303:  begin EndingMass:=PreviousMass; EndingTime:=TimeFromPreviousLine; end
1304:   else begin EndingMass:=Mass; EndingTime:=TimeFromCurrentLine; end;
1305:  if StartingMass<>0 then begin
1306:                          DeltaMass:=EndingMass-StartingMass;
1307:                          DeltaTime:=EndingTime-StartingTime;
1308:                          end else begin DeltaMass:=0; DeltaTime:=0; end;
1309: if DeltaTime<>0 then RateOfMassGain:=(DeltaMass / DeltaTime)*10000 else
1310:   RateOfMassGain:=0;
1311:  if DifferenceB<=DifferenceA then   begin if Finished=False then begin
1312:  WriteLn(Results, TimeStr+', '+FloatToStrF(RateOfMassGain, ffFixed,6,2)+', '+
```

```
1313: IntToStr(TimeFromPreviousLine)+', '+PreviousMassStr+', '+PreviousRealTime);
1314:                 end;
1315:     end  else if Finished=False then begin
1316:      WriteLn(Results, TimeStr+', '+FloatToStrF(RateOfMassGain, ffFixed,6,2)+',
      '+
1317: IntToStr(TimeFromCurrentLine)+', '+MassStr+', '+RealTime);
1318:    end;
1319:  CloseFile(Results);
1320:           end;  //data written to results file
1321:
1322: end;
1323:  MassStr:='';  TimeFromPreviousLine:=0; PreviousMassStr:=''; Str:='';
      CommaCount:=1;
1324:  Total:=0;  BtnMassGainPerInt.enabled:=true; TimeFromCurrentLine:=0;
      Initialise:=true;
1325:  StartingMass:=0; EndingMass:=0; DeltaMass:=0; Finished:=False;
1326:  //all variables cleared and reset as appropriate
1327: end;
1328: end;
1329:
1330:
1331:
1332: procedure TForm1.Button4Click(Sender: TObject);
1333: var str:string; total:integer;
1334: begin
1335: total:=0;
1336:  AssignFile(F, label1.caption+'\'+FileEdit1.text);
1337:   reset(F);
1338:    readLn(F, Str);
1339:    edit3.text:=str;
1340: while Str<>'' do begin Total:=Total+1;
1341:    //EditTotal.text:=inttostr(total);
1342:     //EditTotal.refresh;
1343:  ReadLn(F, Str); end;
1344:   CloseFile(F); //************end of line counter- variable is called
      Total.******************
1345:    Edit3.text:=inttostr(total);
1346:
1347:
1348:
1349: end;
1350:
1351: procedure TForm1.Button5Click(Sender: TObject);
1352: begin
1353: edit6.text:='10 07 98';
1354: edit8.text:='24 08 98';//sets dates for TV
1355: end;
1356:
1357:
1358: //code to condense TB Files from 34 columns to just 17
1359: procedure TForm1.Button6Click(Sender: TObject);
1360: var RealTime, originalFileName, words, resultStr, CurrentDate  :string;
1361:  CommaCounter:integer;
1362:  label 2; label 3;
1363: begin
1364: //code highlighted below is responsible for reading files sequentially
1365: ////////////////////////////////////////////////////////////////////////////
1366: CurrentDate:=edit6.text;
1367:
1368: 2:
1369:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1370:  begin CurrentDate:=NextDayOfYear(CurrentDate); end;
1371:  edit7.text:=label1.caption+'\'+'FeedTBData'+' ['+edit6.text+']';
1372:   edit7.refresh;
1373: edit6.text:=CurrentDate; edit6.refresh;
1374: if FileExists(label1.caption+'\'+'FeedTBData'+' ['+edit6.text+'].txt')
1375:  then begin
1376:
1377: assignFile(F,  label1.caption+'\'+'FeedTBData'+' ['+edit6.text+'].txt');
1378: ////////////////////////////////////////////////////////////////////////////
1379: commaCounter:=0; //allows first column to be read in as default
1380: edit5.text:='data to c:\Temp (Modi Prefix)';
1381: //***replaced by sequential reader code***  AssignFile(results,
      'c:\Temp\Modi'+FileEdit1.text);
```

```
1382: AssignFile(Results, 'c:\Temp\ModiFeedTBData ['+edit6.text+'].txt');
1383: Rewrite(results);
1384:  closefile(results);
1385:  //***replaced by sequential reader code***  AssignFile(F,
      label1.caption+'\'+FileEdit1.text);
1386:  reset(F);
1387:      //this section of code pre-reads the file and calculates the number of
      lines
1388:  readLn(F, Str);
1389:  while Str<>'' do begin Total:=Total+1;
1390:     //EditTotal.text:=inttostr(total);
1391:     //EditTotal.refresh;
1392:  ReadLn(F, Str); end;
1393:   CloseFile(F); //end of line counter- variable is called Total.
1394:   Edit3.text:=inttostr(total);
1395:    Reset(F);
1396: ////////////////////////////////////////////////////////////////////////////
1397: for k:=1 to total do begin //should it be total minus one?
1398: readLn(F, Str);
1399:  StrLength:=Length(Str);
1400:  for i:=1 to StrLength do   begin
1401:   if Str[i]<>',' then if Str[i]<>'*' then  if commaCounter<4 then if
      commaCounter>2 then
1402:
1403:    resultStr:=resultStr+Str[i];
1404:   if commaCounter=0 then if str[i]<>',' then
1405:     resultStr:=resultStr+Str[i]; //in order to read in first column, but not
      comma
1406:
1407:    if str[i]=',' then begin commaCounter:=commaCounter+1;
1408:                             if commaCounter=5 then commaCounter:=1;
1409:                             if commaCounter<4 then
1410:                              if commaCounter>2 then resultStr:=resultStr+';';
1411:                       end;
1412:                             end;
1413:   CommaCounter:=0; //reads in first column
1414: Append(results);
1415:  WriteLn(results, resultStr);
1416:   CloseFile(results);
1417:    ResultStr:='';
1418: end; //of k:=1 to total routine***
1419:     CloseFile(F);
1420:
1421: ////////////////////////////
1422:  end;//of 'if FileExist routine'**********************************
1423:
1424:
1425:
1426: if CurrentDate=edit8.text then goto 3;
1427: goto 2;
1428: 3:
1429: ////////////////////////////
1430: end;
1431:
1432:
1433:
1434: //condenses PerchTBFiles- name specific
1435: procedure TForm1.Button7Click(Sender: TObject);
1436: var RealTime, originalFileName, words, resultStr, CurrentDate  :string;
1437:  CommaCounter:integer;
1438:  label 2; label 3;
1439: begin
1440: //code highlighted below is responsible for reading files sequentially
1441: ////////////////////////////////////////////////////////////////////////////
1442: CurrentDate:=edit6.text;
1443:
1444: 2:
1445:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1446:  begin CurrentDate:=NextDayOfYear(CurrentDate); end;
1447:  edit7.text:=label1.caption+'\'+'PerchTBData'+' ['+edit6.text+']';
1448:   edit7.refresh;
1449: edit6.text:=CurrentDate; edit6.refresh;
1450: if FileExists(label1.caption+'\'+'PerchTBData'+' ['+edit6.text+'].txt')
1451:  then begin
```

```
1452:
1453: assignFile(F,  label1.caption+'\'+'PerchTBData'+' ['+edit6.text+'].txt');
1454: ////////////////////////////////////////////////////////////////////////////
1455: commaCounter:=0; //allows first column to be read in as default
1456: edit5.text:='data to c:\Temp (Modi Prefix)';
1457: //***replaced by sequential reader code***  AssignFile(results,
      'c:\Temp\Modi'+FileEdit1.text);
1458: AssignFile(Results, 'c:\Temp\ModiPerchTBData ['+edit6.text+'].txt');
1459: Rewrite(results);
1460:  closefile(results);
1461:  //***replaced by sequential reader code***  AssignFile(F,
      label1.caption+'\'+FileEdit1.text);
1462:  reset(F);
1463:      //this section of code pre-reads the file and calculates the number of
      lines
1464:  readLn(F, Str);
1465:  while Str<>'' do begin Total:=Total+1;
1466:     //EditTotal.text:=inttostr(total);
1467:     //EditTotal.refresh;
1468:  ReadLn(F, Str); end;
1469:   CloseFile(F); //end of line counter- variable is called Total.
1470:   Edit3.text:=inttostr(total);
1471:    Reset(F);
1472: ////////////////////////////////////////////////////////////////////////////
1473: for k:=1 to total do begin //should it be total minus one?
1474: readLn(F, Str);
1475:  StrLength:=Length(Str);
1476:  for i:=1 to StrLength do   begin
1477:   if Str[i]<>',' then if Str[i]<>'*' then  if commaCounter<4 then if
      commaCounter>2 then
1478:
1479:    resultStr:=resultStr+Str[i];
1480:   if commaCounter=0 then if str[i]<>',' then
1481:     resultStr:=resultStr+Str[i]; //in order to read in first column, but not
      comma
1482:
1483:    if str[i]=',' then begin commaCounter:=commaCounter+1;
1484:                             if commaCounter=5 then commaCounter:=1;
1485:                             if commaCounter<4 then
1486:                              if commaCounter>2 then resultStr:=resultStr+';';
1487:                       end;
1488:                             end;
1489:   CommaCounter:=0; //reads in first column
1490: Append(results);
1491:  WriteLn(results, resultStr);
1492:   CloseFile(results);
1493:    ResultStr:='';
1494: end; //of k:=1 to total routine***
1495:     CloseFile(F);
1496:
1497: ////////////////////////////////
1498:  end;//of 'if FileExist routine'**********************************
1499:
1500:
1501:
1502: if CurrentDate=edit8.text then goto 3;
1503: goto 2;
1504: 3:
1505: ////////////////////////////////
1506: end;
1507:
1508:
1509:
1510: //Code to totalise number of hops on each hop perch for a given day
1511: procedure TForm1.HopsPerPerchButtonClick(Sender: TObject);
1512: var
1513: i, mk1, Starttt :integer; TimeStr, str, CurrentDate:string;
1514: PerchOne, PerchTwo, PerchThree, PerchFour:Integer;
1515: label 1; label 2; label 3;
1516: begin
1517:  PerchOne:=0; PerchTwo:=0; PerchThree:=0; PerchFour:=0;
1518: CurrentDate:=edit6.text;
1519: AssignFile(Results, 'c:\temp\Cumulative Individual Perch data.txt');
1520: rewrite(Results);
```

```
1521: 2:
1522:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1523:  CageNumber:=CageNumber+1; if CageNumber=9 then
1524:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1525:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
       ['+edit6.text+']';
1526:   edit7.refresh;
1527: edit6.text:=CurrentDate; edit6.refresh;
1528:
1529:
1530: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt')
1531:  then begin
1532:
1533: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt');
1534: edit5.text:='data to c:\ParsedPerchData';
1535:
1536: TransformPerchData.default:=true;
1537: //AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
1538:
1539:
1540:   reset(F);
1541: //  Rewrite(times);
1542:  // rewrite(results);
1543:
1544:   //this section of code pre-reads the file and calculates the number of lines.
1545:   readLn(F, Str);
1546:   while Str<>'' do begin Total:=Total+1;
1547:      //EditTotal.text:=inttostr(total);
1548:     //EditTotal.refresh;
1549:   ReadLn(F, Str);
1550:                     end;
1551: CloseFile(F); //end of line counter- no. of lines held in Total.
1552:   Edit3.text:=inttostr(total);
1553:   Reset(F);
1554:   FileListBox1.color:=cllime;filelistbox1.refresh;
1555:   for k:=1 to total do begin //reads through each line of the textfile, this is
       the ROUTINE
1556:      ReadLn(F, Str);
1557:       if Str='' then Str:='0'; //loads current line of textfile into Str
       variable.
1558: if CageNumber=1 then begin//corrects perch numbers to correspond to standard...
1559: if str[1]='3' then PerchOne:=PerchOne+1;
1560:  if str[1]='2' then PerchTwo:=PerchTwo+1;
1561:   if str[1]='4' then PerchThree:=PerchThree+1;
1562:    if str[1]='1' then PerchFour:=PerchFour+1;
1563:                     end;
1564:  if CageNumber=2 then begin
1565:   if str[1]='2' then PerchOne:=PerchOne+1;
1566:    if str[1]='3' then PerchTwo:=PerchTwo+1;
1567:     if str[1]='1' then PerchThree:=PerchThree+1;
1568:      if str[1]='4' then PerchFour:=PerchFour+1;
1569:                       end;
1570:    if CageNumber=3 then begin
1571:     if str[1]='2' then PerchOne:=PerchOne+1;
1572:      if str[1]='1' then PerchTwo:=PerchTwo+1;
1573:       if str[1]='3' then PerchThree:=PerchThree+1;
1574:        if str[1]='4' then PerchFour:=PerchFour+1;
1575:                       end;
1576:       if CageNumber>3 then begin
1577:        if str[1]='1' then PerchOne:=PerchOne+1;
1578:         if str[1]='2' then PerchTwo:=PerchTwo+1;
1579:          if str[1]='3' then PerchThree:=PerchThree+1;
1580:           if str[1]='4' then PerchFour:=PerchFour+1;
1581:                           end;
1582: end; //of 'for k=1 to total' routine...
1583: EditResults.text:=IntToStr(PerchOne)+','+IntToStr(PerchTwo)+','+
1584: IntToStr(PerchThree)+','+IntToStr(PerchFour); editResults.refresh;
1585: Append(results);
1586:  WriteLn(results, floattoStrF(PerchOne/total,  ffFixed, 4, 3)+'; '+
1587:                  floattoStrF(PerchTwo/total,  ffFixed, 4, 3)+'; '+
1588:                  floattoStrF(PerchThree/total,ffFixed, 4, 3)+'; '+
1589:                  floattoStrF(PerchFour/total, ffFixed, 4, 3)+'; '+
```

```
1590:                   IntToStr(CageNumber)+'; '+
1591:                   CurrentDate);
1592:    CloseFile(Results);
1593:    PerchOne:=0; PerchTwo:=0; PerchThree:=0; PerchFour:=0;  total:=0;
1594: End; //of 'if file exists' routine  ...
1595: if CurrentDate=edit8.text then goto 3;
1596: goto 2;
1597: 3:
1598:
1599:
1600: end;
1601:
1602:
1603: procedure TForm1.UpDown1Click(Sender: TObject; Button: TUDBtnType);
1604: begin
1605: edit1.text:=inttostr(strToInt(Edit1.text)+1);
1606:  if edit1.text='7' then edit1.text:='6';
1607: end;
1608:
1609:
1610: ////////////////////////////////////////////////////////////////////////////
1611: procedure TForm1.Button1Click(Sender: TObject);
1612: var
1613: TotalMovements,
1614: PerchOne, PerchTwo, PerchThree, PerchFour:Integer;
1615: i, mk1, Starttt :integer; TimeStr, str, CurrentDate:string;
1616: label 1; label 2; label 3;
1617: begin
1618: PerchOne:=0; PerchTwo:=0; PerchThree:=0; PerchFour:=0;
1619: CurrentDate:=edit6.text;//sets starting date using default value
1620:
1621: 2:
1622:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1623:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
      back to 1 again
1624:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1625:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
      ['+edit6.text+']';
1626:   edit7.refresh;  //displays the file (with filepath) currently being parsed
      (if it exists)
1627: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
1628: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt')
1629: //checks whether file exists or not, if so then the code below is executed
1630:  then begin
1631:
1632: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt');
1633: //F identifies the textfile currently being parsed
1634: edit5.text:='data to c:\ParsedPerchData';
1635: totalPerchMovements:=0; //this variable is used to count total hops per period
1636:
1637: TransformPerchData.default:=true;//makes button the new default control
1638: AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
1639: AssignFile(Results, 'c:\temp\PropHopsPerPerch'+IntToStr(CageNumber)+'
      ['+CurrentDate+'].txt');
1640:  //assigns files for storage of time divisions and the Results Output (Modi
      prefix)
1641:
1642:   reset(F);        //resets file being read
1643:  //  Rewrite(times);  //clears any previous content in times file
1644:       rewrite(results);//clears any previous content in results file
1645:
1646:   //this section of code pre-reads the file and calculates the number of lines.
1647:   readLn(F, Str);
1648:   while Str<>'' do begin Total:=Total+1; //total lines
1649:      //EditTotal.text:=inttostr(total);
1650:     //EditTotal.refresh;
1651:   ReadLn(F, Str);
1652:                   end;
1653: CloseFile(F); //end of line counter- no. of lines held in Total.
1654:   Edit3.text:=inttostr(total); //displays total lines
1655:   Reset(F);   //resets file being read again
1656:   FileListBox1.color:=cllime;filelistbox1.refresh; //causes FileListBox to
```

```
    flash
1657:    for k:=1 to total do begin //reads through each line of the textfile, this is
    the MAIN ROUTINE
1658:       ReadLn(F, Str); //loads current line of textfile into Str variable.
1659:
1660:       for i:=1 to 16 do begin // reads each character of string in turn.
1661:       //only reads to number 16 to avoid reading in the third comma, c.f.
    BalanceFileParser
1662:          if Str[i]=',' then begin if flag=false then begin mk1:=i; flag:=true
1663:                                                          end
1664:          else begin
1665:               mk2:=i; flag:=false; //comma positions located in mk1 and mk2
1666:               end;           end;
1667:                                 end;
1668:               //finds positions of the first two commas in line (only)
1669:          CommaGap:=mk2-mk1; //calculates the distance between the two commas.
1670: if CommaGap=9 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]);
1671: if CommaGap=10 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]);
1672: if TimeStr<>'' then Time:=strToInt(TimeStr); //this is the time in seconds.
1673: //*****************************************************************************
1674: 1:
1675:    with TimesIni do begin //TimesIni is the .ini file containing the current
    period settings
1676:       Start:=ReadInteger('SecondGaps',
    'Interval'+Inttostr(intervalCounter)+'Start', -1);
1677:       Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
    -1);
1678:       //Start and Endd are variables containing start and end of interval in
    seconds.
1679:                      end;
1680: If Time>=Start then if Time<=Endd then begin
1681:
1682:          if Str='' then Str:='0'; //loads current line of textfile into Str
    variable.
1683: if CageNumber=1 then begin//corrects perch numbers to correspond to standard...
1684: if str[1]='3' then PerchOne:=PerchOne+1;
1685:  if str[1]='2' then PerchTwo:=PerchTwo+1;
1686:   if str[1]='4' then PerchThree:=PerchThree+1;
1687:    if str[1]='1' then PerchFour:=PerchFour+1;
1688:                       end;
1689:  if CageNumber=2 then begin
1690:   if str[1]='2' then PerchOne:=PerchOne+1;
1691:    if str[1]='3' then PerchTwo:=PerchTwo+1;
1692:     if str[1]='1' then PerchThree:=PerchThree+1;
1693:      if str[1]='4' then PerchFour:=PerchFour+1;
1694:                        end;
1695:    if CageNumber=3 then begin
1696:     if str[1]='2' then PerchOne:=PerchOne+1;
1697:      if str[1]='1' then PerchTwo:=PerchTwo+1;
1698:       if str[1]='3' then PerchThree:=PerchThree+1;
1699:        if str[1]='4' then PerchFour:=PerchFour+1;
1700:                       end;
1701:       if CageNumber>3 then begin
1702:        if str[1]='1' then PerchOne:=PerchOne+1;
1703:         if str[1]='2' then PerchTwo:=PerchTwo+1;
1704:          if str[1]='3' then PerchThree:=PerchThree+1;
1705:           if str[1]='4' then PerchFour:=PerchFour+1;
1706:                          end;
1707:
1708:    TotalPerchMovements:=TotalPerchMovements+1;
1709:    PerchFlag:=true; //signifies that there was a perch movement during current
    period
1710:                                      end else
1711:    begin
1712:      if PerchFlag=false then begin//if no movements occured then do this
1713:        Append(Results);
1714:        WriteLn(Results, '0; 0; 0; 0'); //writes a zero in as result
1715:        CloseFile(Results);
1716:        IntervalCounter:=IntervalCounter+1;//ups IntervalCounter by one, next
    period
1717:        goto 1;  //sends execution back to see if present perch movement fits
    within next period
1718:                                end          else
1719:         begin  //this code is executed if there are PerchMovements within
```

```
    current period
1720:           TotalMovements:=PerchOne+PerchTwo+PerchThree+PerchFour;
1721:            Append(Results);
1722:              WriteLn(Results, FloatToStrF(PerchOne/TotalMovements, ffFixed, 5,
    4)+'; '+
1723:            FloatToStrF(PerchTwo/TotalMovements, ffFixed, 5, 4)+'; '+
1724:            FloatToStrF(PerchThree/TotalMovements, ffFixed, 5, 4)+'; '+
1725:            FloatToStrF(PerchFour/TotalMovements, ffFixed, 5, 4));//log total
    perch movements within period
1726: PerchOne:=0; PerchTwo:=0; PerchThree:=0; PerchFour:=0; TotalMovements:=0;
1727:            CloseFile(Results);
1728:            TotalPerchMovements:=0;   //zeroes this variable ready for next
    count
1729:            IntervalCounter:=IntervalCounter+1; //ups Interval Counter by one,
    to next period
1730:            PerchFlag:=False; //resets this flag
1731:            goto 1;  //sends execution back to label 1 again
1732:          end;
1733:     end;
1734: //*****************************************************************************
1735: end; //end of 'for k:=1 to total do' routine.
1736:
1737: IntervalCounter:=IntervalCounter+1;//having read to end of file, this section
    fills in blank values so that output files have a uniform number of lines
1738:   if PerchFlag=true then begin//if flag is true then there is remaining data to
    be written
1739:      Append(Results);
1740:      TotalMovements:=PerchOne+PerchTwo+PerchThree+PerchFour;
1741:      WriteLn(Results, FloatToStrF(PerchOne/TotalMovements, ffFixed, 5, 4)+'; '+
1742:       FloatToStrF(PerchTwo/TotalMovements, ffFixed, 5, 4)+'; '+
1743:        FloatToStrF(PerchThree/TotalMovements, ffFixed, 5, 4)+'; '+
1744:         FloatToStrF(PerchFour/TotalMovements, ffFixed, 5, 4));
1745: PerchOne:=0; PerchTwo:=0; PerchThree:=0; PerchFour:=0;  TotalMovements:=0;
1746:      CloseFile(Results);
1747:      PerchFlag:=False;
1748:
1749: CloseFile(F);//closes current data file as it has been read to the end now...
1750:      while start<>-1 do begin //while there are still periods remaining
1751:      with TimesIni do
1752:      Start:=ReadInteger('SecondGaps',
    'Interval'+intToStr(IntervalCounter)+'Start', -1);
1753:      IntervalCounter:=IntervalCounter+1;
1754:      Append(Results);
1755:      if start<>-1 then WriteLn(Results, '0; 0; 0; 0');
1756:       TotalMovements:=0;  //continues to add blank lines to ensure uniformity
1757:      CloseFile(results);
1758:                          end;
1759:
1760: PerchFlag:=false;//ensures this flag is set to false at this stage
1761: intervalCounter:=0;
1762:
1763:
1764:   mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
    IntervalCheck:=False;
1765:   time:=0;  //sets relevant variables back to default values ready for read of
    next file
1766:
1767:  // CloseFile(Times);
1768:   Total:=0; IntervalCounter:=0; TotalPerchMovements:=0; Start:=0; Endd:=0;
    //more resetting
1769:        FileListBox1.color:=clwindow; //briefly alters color of FileListBox to
    denote end of read
1770:
1771:       end;//of 'if FileExist routine'**********************************
1772:
1773:
1774: End;
1775: if CurrentDate=edit8.text then goto 3;//if CurrentDate is finish date then this
    code stops
1776: goto 2; //else it goes back to label 2 to begin reading next file in the
    sequence
1777: 3:
1778:
1779: end;
```

```
1780:
1781:
1782:
1783:
1784: procedure TForm1.Button2Click(Sender: TObject);
1785: begin
1786: edit6.text:='14 03 98';
1787: edit8.text:='21 04 98';//sets dates for MA2
1788: end;
1789:
1790: procedure TForm1.Button3Click(Sender: TObject);
1791: begin
1792: edit6.text:='31 01 98';
1793: edit8.text:='10 03 98';//sets dates for LRE2
1794: end;
1795:
1796: procedure TForm1.Button8Click(Sender: TObject);
1797: begin
1798: edit6.text:='14 12 98';
1799: edit8.text:='05 02 99';//sets dates for FA
1800: end;
1801:
1802: procedure TForm1.Button9Click(Sender: TObject);
1803: begin
1804: edit6.text:='17 02 99';
1805: edit8.text:='19 03 99';//sets dates for FI
1806: end;
1807:
1808: procedure TForm1.Button10Click(Sender: TObject);
1809: begin
1810: edit6.text:='31 05 99';
1811: edit8.text:='03 07 99';//sets dates for MA3
1812: end;
1813:
1814: procedure TForm1.ParseTemperatureFileClick(Sender: TObject);
1815: var
1816: counter:integer;
1817: begin
1818: filelistbox1.color:=cllime; filelistbox1.refresh;
1819: counter:=59;
1820: assignfile(results, 'c:\temp\parsedTempFile.txt');
1821:  rewrite(results);
1822:  AssignFile(F, label1.caption+'\'+FileEdit1.text);
1823:   reset(F);
1824:    readLn(F, Str);
1825: while str<>'' do begin
1826: readLn(F, Str);
1827:   counter:=counter+1;
1828:    if counter=300 then begin
1829:     counter:=0;
1830:      append(results);
1831:       writeLn(results, str);
1832:         closefile(results);
1833:         end;
1834: end;
1835: filelistbox1.color:=clwhite;
1836: end;
1837:
1838: procedure TForm1.Button11Click(Sender: TObject);
1839: var
1840: TotalDuration:integer;
1841: i, mk1, Starttt :integer; TimeStr, str, CurrentDate:string;
1842: label 1; label 2; label 3;
1843: begin
1844: CurrentDate:=edit6.text;//sets starting date using default value
1845:
1846: 2:
1847:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1848:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
      back to 1 again
1849:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1850:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
      ['+edit6.text+']';
1851:   edit7.refresh;  //displays the file (with filepath) currently being parsed
```

```
      (if it exists)
1852: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
1853: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt')
1854: //checks whether file exists or not, if so then the code below is executed
1855:  then begin
1856:
1857: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
      ['+edit6.text+'].txt');
1858: //F identifies the textfile currently being parsed
1859: edit5.text:='data to c:\ParsedPerchData';
1860: totalPerchMovements:=0; //this variable is used to count total hops per period
1861:
1862: TransformPerchData.default:=true;//makes button the new default control
1863: AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
1864: AssignFile(Results, 'c:\temp\ModiPerch'+IntToStr(CageNumber)+'
      ['+CurrentDate+'].txt');
1865:  //assigns files for storage of time divisions and the Results Output (Modi
      prefix)
1866:
1867:   reset(F);        //resets file being read
1868:   Rewrite(times);  //clears any previous content in times file
1869:   rewrite(results);//clears any previous content in results file
1870:
1871:   //this section of code pre-reads the file and calculates the number of lines.
1872:   readLn(F, Str);
1873:   while Str<>'' do begin Total:=Total+1; //total lines
1874:      //EditTotal.text:=inttostr(total);
1875:     //EditTotal.refresh;
1876:   ReadLn(F, Str);
1877:                     end;
1878: CloseFile(F); //end of line counter- no. of lines held in Total.
1879:   Edit3.text:=inttostr(total); //displays total lines
1880:   Reset(F);   //resets file being read again
1881:   FileListBox1.color:=cllime;filelistbox1.refresh; //causes FileListBox to
      flash
1882:   for k:=1 to total do begin //reads through each line of the textfile, this is
      the MAIN ROUTINE
1883:       ReadLn(F, Str); //loads current line of textfile into Str variable.
1884:       if k=1 then Str:='1, 10000, 00:00:10, 9999';
1885:  for i:=1 to Length(str) do begin// reads each character of whole string in
      turn.
1886: if str[i]=',' then begin
1887:    if Commaflag=1 then begin mk1:=i; Commaflag:=2 end else
1888:     if Commaflag=2 then begin mk2:=i; Commaflag:=3 end else
1889:      if Commaflag=3 then begin mk3:=i; Commaflag:=1 end;
1890:                     end;  end; //above routine finds the positions of the commas
1891:       CommaGap:=mk2-mk1; //calculates the distance between the two commas.
1892: if CommaGap=9 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]);
1893: if CommaGap=10 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]);
1894: if TimeStr<>'' then Time:=strToInt(TimeStr); //this is the time in seconds.
1895: duration:='';//sets duration to a null string.
1896: for i:=mk3+2 to Length(Str) do
1897:  duration:=duration+Str[i];
1898: //*****************************************************************************
1899: 1:
1900:   with TimesIni do begin //TimesIni is the .ini file containing the current
      period settings
1901:      Start:=ReadInteger('SecondGaps',
      'Interval'+Inttostr(intervalCounter)+'Start', -1);
1902:      Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
      -1);
1903:      //Start and Endd are variables containing start and end of interval in
      seconds.
1904:                     end;
1905: If Time>=Start then if Time<=Endd then begin
1906:      TotalPerchMovements:=TotalPerchMovements+1; //sums total hops
1907:      TotalDuration:=TotalDuration+StrtoInt(Duration);      //sums total time on
      perches
1908:      PerchFlag:=true; //signifies that there was a perch movement during current
      period
1909:                                        end else
1910:      begin
1911:        if PerchFlag=false then begin//if no movements occured then do this
```

```
1912:         Append(Results);
1913:         WriteLn(Results, 0); //writes a zero in as result
1914:         CloseFile(Results);
1915:         IntervalCounter:=IntervalCounter+1;//ups IntervalCounter by one, next
      period
1916:         goto 1;  //sends execution back to see if present perch movement fits
      within next period
1917:                                 end           else
1918:           begin  //this code is executed if there are PerchMovements within
      current period
1919:             Append(Results);
1920:             WriteLn(Results, TotalDuration/TotalPerchMovements);//log total
      perch movements within period
1921:             CloseFile(Results);
1922:             TotalPerchMovements:=0;   //zeroes this variable ready for next
      count
1923:             TotalDuration:=0;         //zeroes this also ready for next count
1924:             IntervalCounter:=IntervalCounter+1; //ups Interval Counter by one,
      to next period
1925:             PerchFlag:=False; //resets this flag
1926:             goto 1;  //sends execution back to label 1 again
1927:           end;
1928:     end;
1929: //*****************************************************************************
1930: end; //end of 'for k:=1 to total do' routine.
1931:
1932: IntervalCounter:=IntervalCounter+1;//having read to end of file, this section
      fills in blank values so that output files have a uniform number of lines
1933:   if PerchFlag=true then begin//if flag is true then there is remaining data to
      be written
1934:       Append(Results);
1935:       WriteLn(Results, TotalDuration/TotalPerchMovements);
1936:       CloseFile(Results);
1937:       PerchFlag:=False;
1938:
1939: CloseFile(F);//closes current data file as it has been read to the end now...
1940:       while start<>-1 do begin //while there are still periods remaining
1941:       with TimesIni do
1942:       Start:=ReadInteger('SecondGaps',
      'Interval'+intToStr(IntervalCounter)+'Start', -1);
1943:       IntervalCounter:=IntervalCounter+1;
1944:       Append(Results);
1945:       if start<>-1 then WriteLn(Results, 0);//continues to add blank lines to
      ensure uniformity
1946:       CloseFile(results);
1947:                           end;
1948:
1949: PerchFlag:=false;//ensures this flag is set to false at this stage
1950: intervalCounter:=0;
1951:   for i:=1 to 80 do begin //this code creates the file containing current
      period start times
1952:   //it runs the code 80 times, this is arbitrary and exceeds maximum no. of
      possible lines
1953:   with TimesIni do
1954:   Start:=ReadInteger('SecondGaps',
      'Interval'+inttostr(IntervalCounter)+'Start', -1);
1955:   if start<>-1 then WriteLn(Times, (Inttostr(Start div 60)));//divides by 60
      hence minutes
1956:   IntervalCounter:=IntervalCounter+1;
1957:                         end;
1958:   mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
      IntervalCheck:=False;
1959:   time:=0;  //sets relevant variables back to default values ready for read of
      next file
1960:
1961:   CloseFile(Times);
1962:   Total:=0; IntervalCounter:=0; TotalPerchMovements:=0; Start:=0; Endd:=0;
      //more resetting
1963:       TotalDuration:=0;
1964:         FileListBox1.color:=clwindow; //briefly alters color of FileListBox to
      denote end of read
1965:
1966:       end;//of 'if FileExist routine'*********************************
1967:
```

```
1968:
1969: End;
1970: if CurrentDate=edit8.text then goto 3;//if CurrentDate is finish date then this
       code stops
1971: goto 2; //else it goes back to label 2 to begin reading next file in the
       sequence
1972: 3:
1973:
1974: end;
1975:
1976:
1977:
1978:
1979: procedure TForm1.BtnMeanHopGapClick(Sender: TObject);
1980: label 1; label 2; label 3;
1981: var CurrentDate, PreviousStr, PreviousDuration : String; TotalHopGap,
       totaliser:extended;
1982:
1983: begin
1984:  2:
1985:   TotalHopGap:=0;
1986:   totaliser:=0;
1987: CurrentDate:=edit6.text;//sets starting date using default value
1988:
1989:
1990:  FileListBox1.color:=clwhite;filelistbox1.refresh;
1991:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
       back to 1 again
1992:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
1993:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
       ['+edit6.text+']';
1994:   edit7.refresh;  //displays the file (with filepath) currently being parsed
       (if it exists)
1995: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
1996: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt')
1997: //checks whether file exists or not, if so then the code below is executed
1998:  then begin
1999: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
       ['+edit6.text+'].txt');
2000:
2001: edit5.text:='Data to c:\ParsedPerchData, with Alt. prefix';//indicates path to
       results
2002: intervalCounter:=0; DisplayValue:=0; TotalDisplayValue:=0;  //resets some
       variables
2003:  ButtonAwardPerchScore.default:=true;//makes this the default control
2004: //AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
2005: AssignFile(Results, 'c:\ParsedPerchData\Alt1.Perch '+IntToStr(CageNumber)+'
       ['+CurrentDate+'].txt');;
2006: // AssignFile(F, label1.caption+'\'+FileEdit1.text);
2007:   reset(F); //resets source file ready for reading
2008:   rewrite(results);//clears results file ready for writing
2009: //  Rewrite(times);
2010:   //this section of code pre-reads the file and calculates the number of lines.
2011:   readLn(F, Str);
2012:    FileListBox1.color:=cllime;filelistbox1.refresh;//flashes list box to denote
       finish
2013:   while Str<>'' do begin Total:=Total+1;
2014:   ReadLn(F, Str);
2015:                    end;
2016: CloseFile(F); //end of line counter- no. of lines held in Total.
2017:   Edit3.text:=inttostr(total);//displays total no. of lines
2018:   Reset(F); //resets data textfile for next read of values
2019: //*****************************************************************************
2020:   for k:=1 to total-1 do begin //reads through each line of the textfile, this
       is the ROUTINE
2021:  PreviousStr:=Str;
2022:      ReadLn(F, Str); //loads current line of textfile into Str variable.
2023:      CommaFlag:=1;//sets default value
2024: if Str='' then Str:=PreviousStr;
2025:      for i:=1 to Length(str) do begin// reads each character of whole string in
       turn.
2026: if str[i]=',' then begin
2027:    if Commaflag=1 then begin mk1:=i; Commaflag:=2 end else
```

```
2028:     if Commaflag=2 then begin mk2:=i; Commaflag:=3 end else
2029:      if Commaflag=3 then begin mk3:=i; Commaflag:=1 end;
2030:                    end;  end; //above routine finds the positions of the commas
2031:
2032:
2033:        CommaGap:=mk2-mk1; //calculates the distance between first two commas.
2034:
2035: if CommaGap=9 then
       timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]+str[9]+str[10]);
2036: if CommaGap=10 then
       timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]+str[9]+str[10]+str[11]);
2037: PreviousTime:=TimeA;//keeps a running record of previous time value
2038: if TimeStr<>'' then TimeA:=strToInt(TimeStr); //this is the time in
       milliseconds.
2039: HopGap:=TimeA-PreviousTime; //Time in milliseconds between present and previous
       hops
2040: if HopGap<=0 then HopGap:=1;//prevents divide by zero errors, 1ms is a
       negligible value
2041:
2042: PreviousPerchID:=PerchID;//keeps a running note of which perch was active
2043: PerchID:=str[1]; //Identity of perch which is depressed
2044: if duration<>'' then PreviousDuration:=Duration else PreviousDuration:='1';
2045: duration:=''; //clears durations string prior to writing a new value into it
2046: for j:=mk3+2 to Length(Str) do
2047:  duration:=duration+Str[j];//gives string 'Duration' with duration in Msecs
2048: Time:=abs(TimeA div 1000); //converts time into seconds in readiness for next
       section of code
2049: //*****************************************************************************
2050: 1:
2051:   with TimesIni do begin//refers to .ini file for start and finish times
2052:     Start:=ReadInteger('SecondGaps',
      'Interval'+Inttostr(intervalCounter)+'Start', -1);
2053:     Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
      -1);
2054:                    end;
2055: If Time>=Start then if Time<Endd then begin
2056: //*****************************************************************************
2057:   if StrToInt(Duration)=0 then Duration:='1';
2058:    if cageNumber=7 then
2059:          {do nothing} else  begin
2060: TotalHopGap:=TotalHopGap+(HopGap)-StrToInt(PreviousDuration);
2061:  totaliser:=totaliser+1;
2062:   PerchFlag:=true;           end;
2063:
2064:                                      end else
2065:     begin
2066:       if PerchFlag=false then begin
2067:         Append(Results);
2068:         WriteLn(Results, 0);//if no data for that period then a zero is awarded
2069:         CloseFile(Results);
2070:         IntervalCounter:=IntervalCounter+1;//clicks forward to the next
       interval
2071:         goto 1;
2072:                              end          else
2073:           begin
2074:            Append(Results);
2075:            WriteLn(Results, floattostrF((TotalHopGap/Totaliser), ffFixed, 6,
       5));
2076:            CloseFile(Results); //writes the perch display index into the
       results textfile
2077:            Totaliser:=0;
2078:            TotalHopGap:=0;
2079:            IntervalCounter:=IntervalCounter+1; //clicks forward onto next
       interval
2080:            PerchFlag:=False;
2081:            goto 1; //sends execution point back to 1
2082:           end;
2083:     end;
2084: //*****************************************************************************
2085: end; //end of 'for k:=1 to total do' routine.
2086:
2087: IntervalCounter:=IntervalCounter+1;//clicks forward to next interval
2088:   if PerchFlag=true then begin
2089:      Append(Results);
```

```
2090:      WriteLn(Results, floattostrF((TotalHopGap/Totaliser), ffFixed, 6, 5));
2091:      CloseFile(Results);//writes remaining data to results, if it is present
2092:      PerchFlag:=False;
2093:
2094: CloseFile(F);
2095:      while start<>-1 do begin
2096:      with TimesIni do
2097:      Start:=ReadInteger('SecondGaps',
     'Interval'+intToStr(IntervalCounter)+'Start', -1);
2098:      IntervalCounter:=IntervalCounter+1;
2099:      Append(Results);
2100:      if start<>-1 then WriteLn(Results, 0);//fills in zero values if there is
     no more data
2101:      CloseFile(results);
2102:                        end;
2103:
2104: PerchFlag:=false;
2105: intervalCounter:=0;
2106:
2107:
2108:   mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
     IntervalCheck:=False;
2109:   timeA:=0;
2110:
2111:   Total:=0; IntervalCounter:=0; TotalDisplayValue:=0; Start:=0; Endd:=0;
2112:        FileListBox1.color:=clwindow;
2113:     //   edit10.text:=floattostr(totalhopgap/totaliser);
2114:  end;
2115:   end;
2116: if CurrentDate=edit8.text then goto 3;
2117:  goto 2;
2118:   3:
2119:  end;
2120:
2121: procedure TForm1.Button12Click(Sender: TObject);
2122: var
2123: TotalDuration:integer;
2124: i, mk1, Starttt :integer; TimeStr, str, CurrentDate:string;
2125: label 1; label 2; label 3;
2126: begin
2127: CurrentDate:=edit6.text;//sets starting date using default value
2128:
2129: 2:
2130:  FileListBox1.color:=clwhite;filelistbox1.refresh;
2131:  CageNumber:=CageNumber+1; if CageNumber=9 then //if cageNumber = 9 then revert
     back to 1 again
2132:  begin CageNumber:=1; CurrentDate:=NextDayOfYear(CurrentDate); end;
2133:  edit7.text:=label1.caption+'\'+'Perch '+IntToStr(CageNumber)+'
     ['+edit6.text+']';
2134:   edit7.refresh;  //displays the file (with filepath) currently being parsed
     (if it exists)
2135: edit6.text:=CurrentDate; edit6.refresh; //displays the current date being read
2136: if FileExists(label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
     ['+edit6.text+'].txt')
2137: //checks whether file exists or not, if so then the code below is executed
2138:  then begin
2139:
2140: assignFile(F,  label1.caption+'\'+'Perch'+IntToStr(CageNumber)+'
     ['+edit6.text+'].txt');
2141: //F identifies the textfile currently being parsed
2142: edit5.text:='data to c:\ParsedPerchData';
2143: totalPerchMovements:=0; //this variable is used to count total hops per period
2144:
2145: TransformPerchData.default:=true;//makes button the new default control
2146: AssignFile(times, 'c:\ParsedPerchData\Current Time Divisions (Mins).txt');
2147: AssignFile(Results, 'c:\temp\ModiPerch'+IntToStr(CageNumber)+'
     ['+CurrentDate+'].txt');
2148:  //assigns files for storage of time divisions and the Results Output (Modi
     prefix)
2149:
2150:   reset(F);        //resets file being read
2151:   Rewrite(times);  //clears any previous content in times file
2152:   rewrite(results);//clears any previous content in results file
2153:
```

```
2154:   //this section of code pre-reads the file and calculates the number of lines.
2155:   readLn(F, Str);
2156:   while Str<>'' do begin Total:=Total+1; //total lines
2157:      //EditTotal.text:=inttostr(total);
2158:     //EditTotal.refresh;
2159:   ReadLn(F, Str);
2160:                     end;
2161: CloseFile(F); //end of line counter- no. of lines held in Total.
2162:   Edit3.text:=inttostr(total); //displays total lines
2163:   Reset(F);   //resets file being read again
2164:   FileListBox1.color:=cllime;filelistbox1.refresh; //causes FileListBox to
      flash
2165:   for k:=1 to total do begin //reads through each line of the textfile, this is
      the MAIN ROUTINE
2166:      ReadLn(F, Str); //loads current line of textfile into Str variable.
2167:      if k=1 then Str:='1, 10000, 00:00:10, 9999';
2168:  for i:=1 to Length(str) do begin// reads each character of whole string in
      turn.
2169: if str[i]=',' then begin
2170:    if Commaflag=1 then begin mk1:=i; Commaflag:=2 end else
2171:     if Commaflag=2 then begin mk2:=i; Commaflag:=3 end else
2172:      if Commaflag=3 then begin mk3:=i; Commaflag:=1 end;
2173:                     end;  end; //above routine finds the positions of the commas
2174:      CommaGap:=mk2-mk1; //calculates the distance between the two commas.
2175: if CommaGap=9 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]);
2176: if CommaGap=10 then timeStr:=concat(str[4]+str[5]+str[6]+str[7]+str[8]);
2177: if TimeStr<>'' then Time:=strToInt(TimeStr); //this is the time in seconds.
2178: duration:='';//sets duration to a null string.
2179: for i:=mk3+2 to Length(Str) do
2180:  duration:=duration+Str[i];
2181: //*****************************************************************************
2182: 1:                                                                    .
2183:   with TimesIni do begin //TimesIni is the .ini file containing the current
      period settings
2184:      Start:=ReadInteger('SecondGaps',
      'Interval'+Inttostr(intervalCounter)+'Start', -1);
2185:      Endd:=ReadInteger('SecondGaps', 'Interval'+IntToStr(IntervalCounter)+'End',
      -1);
2186:      //Start and Endd are variables containing start and end of interval in
      seconds.
2187:                     end;
2188: If Time>=Start then if Time<=Endd then begin
2189:      TotalPerchMovements:=TotalPerchMovements+1; //sums total hops
2190:      TotalDuration:=TotalDuration+StrtoInt(Duration);      //sums total time on
      perches
2191:      PerchFlag:=true; //signifies that there was a perch movement during current
      period
2192:                                          end else
2193:     begin
2194:       if PerchFlag=false then begin//if no movements occured then do this
2195:         Append(Results);
2196:         WriteLn(Results, 0); //writes a zero in as result
2197:         CloseFile(Results);
2198:         IntervalCounter:=IntervalCounter+1;//ups IntervalCounter by one, next
      period
2199:         goto 1;  //sends execution back to see if present perch movement fits
      within next period
2200:                                  end           else
2201:          begin  //this code is executed if there are PerchMovements within
      current period
2202:            Append(Results);
2203:            WriteLn(Results, TotalDuration);//log total perch movements within
      period
2204:            CloseFile(Results);
2205:            TotalPerchMovements:=0;   //zeroes this variable ready for next
      count
2206:            TotalDuration:=0;         //zeroes this also ready for next count
2207:            IntervalCounter:=IntervalCounter+1; //ups Interval Counter by one,
      to next period
2208:            PerchFlag:=False; //resets this flag
2209:            goto 1;  //sends execution back to label 1 again
2210:          end;
2211:     end;
2212: //*****************************************************************************
```

```
2213: end; //end of 'for k:=1 to total do' routine.
2214:
2215: IntervalCounter:=IntervalCounter+1;//having read to end of file, this section
       fills in blank values so that output files have a uniform number of lines
2216:   if PerchFlag=true then begin//if flag is true then there is remaining data to
       be written
2217:       Append(Results);
2218:       WriteLn(Results, TotalDuration);
2219:       CloseFile(Results);
2220:       PerchFlag:=False;
2221:
2222: CloseFile(F);//closes current data file as it has been read to the end now...
2223:       while start<>-1 do begin //while there are still periods remaining
2224:       with TimesIni do
2225:       Start:=ReadInteger('SecondGaps',
       'Interval'+intToStr(IntervalCounter)+'Start', -1);
2226:       IntervalCounter:=IntervalCounter+1;
2227:       Append(Results);
2228:       if start<>-1 then WriteLn(Results, 0);//continues to add blank lines to
       ensure uniformity
2229:       CloseFile(results);
2230:                         end;
2231:
2232: PerchFlag:=false;//ensures this flag is set to false at this stage
2233: intervalCounter:=0;
2234:   for i:=1 to 80 do begin //this code creates the file containing current
       period start times
2235:   //it runs the code 80 times, this is arbitrary and exceeds maximum no. of
       possible lines
2236:   with TimesIni do
2237:   Start:=ReadInteger('SecondGaps',
       'Interval'+inttostr(IntervalCounter)+'Start', -1);
2238:   if start<>-1 then WriteLn(Times, (Inttostr(Start div 60)));//divides by 60
       hence minutes
2239:   IntervalCounter:=IntervalCounter+1;
2240:                     end;
2241:   mk1:=0;  str:='';  timeStr:='';  Flag:=False;  PerchFlag:=false;
       IntervalCheck:=False;
2242:   time:=0;  //sets relevant variables back to default values ready for read of
       next file
2243:
2244:   CloseFile(Times);
2245:   Total:=0; IntervalCounter:=0; TotalPerchMovements:=0; Start:=0; Endd:=0;
       //more resetting
2246:       TotalDuration:=0;
2247:         FileListBox1.color:=clwindow; //briefly alters color of FileListBox to
       denote end of read
2248:
2249:       end;//of 'if FileExist routine'**********************************
2250:
2251:
2252: End;
2253: if CurrentDate=edit8.text then goto 3;//if CurrentDate is finish date then this
       code stops
2254: goto 2; //else it goes back to label 2 to begin reading next file in the
       sequence
2255: 3:
2256:
2257: end;
2258:
2259:
2260: initialization
2261: total:=0; CommaGapA:=0; CommaGapB:=0; CommaGapC:=0; Mk1:=0; Mk2:=0; Mk3:=0;
       Mk4:=0;
2262: CommaCount:=1;  PreviousAverage:='000';  PreviousMass:=0; PreviousTime:=0;
2263: initialise:=true;  IntervalTimeCheck:=0;  TimeFromCurrentLine:=0;
       Finished:=False;
2264: TimeFromPreviousLine:=0;    CageNumber:=0;  CommaFlag:=1;
2265: end.
```

Appendix 3

Full listing of programming code (in Delphi/ Object Pascal) for the **DawnDusk** program described in Chapter 2.

```
unit Unit1;

interface

uses
  SysUtils, WinTypes, WinProcs, Messages, Classes, Graphics, Controls,
  Forms, Dialogs, Menus, ExtCtrls, StdCtrls, unit2;

type
  TForm1 = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    RG1: TRadioGroup;
    Edit2: TEdit;
    Edit3: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Button1: TButton;
    Timer1: TTimer;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    ReselectParameters1: TMenuItem;
    Exit1: TMenuItem;
    About1: TMenuItem;
    Edit1: TEdit;
    ClearLogFile1: TMenuItem;
    ToggleAll: TButton;
    Panel3: TPanel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Panel4: TPanel;
    Button3: TButton;
    Button4: TButton;
    Button5: TButton;
    Button6: TButton;
    Button7: TButton;
    Button8: TButton;
    Button9: TButton;
    Button10: TButton;
    Button11: TButton;
    SH: TMenuItem;
    Panel5: TPanel;
    Edit4: TEdit;
    Edit5: TEdit;
    Label23: TLabel;
    Label24: TLabel;
    EditCutOutTemp: TEdit;
    Label25: TLabel;
    Panel6: TPanel;
    Panel7: TPanel;
    Label26: TLabel;
    Label27: TLabel;
    Label28: TLabel;
    Label29: TLabel;
    BtnSetFansEtc: TButton;
    Panel8: TPanel;
```

```
75:     EditChan1: TEdit;
76:     EditChan2: TEdit;
77:     Label30: TLabel;
78:     Label31: TLabel;
79:     TempTimer: TTimer;
80:     Button2: TButton;
81:     Button12: TButton;
82:     EditTemp: TEdit;
83:     Label32: TLabel;
84:     Label33: TLabel;
85:     Label34: TLabel;
86:     Label35: TLabel;
87:     Button13: TButton;
88:     Label36: TLabel;
89:     Label37: TLabel;
90:     Button14: TButton;
91:     procedure Timer1Timer(Sender: TObject);
92:     procedure Exit1Click(Sender: TObject);
93:     procedure ReselectParameters1Click(Sender: TObject);
94:     procedure Button1Click(Sender: TObject);
95:     procedure About1Click(Sender: TObject);
96:     procedure ClearLogFile1Click(Sender: TObject);
97:     procedure ToggleAllClick(Sender: TObject);
98:     procedure Button3Click(Sender: TObject);
99:     procedure Button4Click(Sender: TObject);
100:    procedure Button5Click(Sender: TObject);
101:    procedure Button6Click(Sender: TObject);
102:    procedure Button7Click(Sender: TObject);
103:    procedure Button8Click(Sender: TObject);
104:    procedure Button9Click(Sender: TObject);
105:    procedure Button10Click(Sender: TObject);
106:    procedure Button11Click(Sender: TObject);
107:    procedure SHClick(Sender: TObject);
108:    procedure BtnSetFansEtcClick(Sender: TObject);
109:    procedure TempTimerTimer(Sender: TObject);
110:    procedure Button2Click(Sender: TObject);
111:    procedure Button12Click(Sender: TObject);
112:    procedure EditTempChange(Sender: TObject);
113:    procedure Button13Click(Sender: TObject);
114:    procedure Button14Click(Sender: TObject);
115:  private
116:    { Private declarations }
117:  public
118:    { Public declarations }
119:  end;
120:
121: var
122:   Form1: TForm1;
123:   GlobalTime, beginDawn, beginDusk, Daylength, DawnDuskLength, LightInterval,
124:   recdusk, Timecheck, SealedLightsOn, SealedLightsOff, Status :longint;
125:   h:integer; Dawn, SealedLightDawn, SealedLightDusk:TDateTime;
126:   a: Textfile;
127:   x, i : integer;
128: relaystatusA, relaystatusB: longint;
129: relaystatusstrA, relaystatusstrB: string;
130: //variables linked to temperature monitoring functions
131: Opened: Boolean;    //Status of communications between PC and TC-08
132: ComPort: Integer;      //designates the Comm Port used
133: TempFile: textfile; //textfile into which to write data
134:   s       : string [10];
135:   ok , FansOn, RoomFansOn     : boolean;
136:   temp    : longint;
137:   ftemp, Temp1, Temp2  : real;
138: data1, data2 : integer;
139:
140: implementation
141:
142: {$R *.DFM}
143:
144:
145:
146: function registerBoardEx(CardNumber:integer):integer; stdcall; external
     'Dio_tc32.dll';
147: function DIOgetDataEx(h, chip, chan:integer; data:Pbyte):integer; stdcall;
```

```
     external 'Dio_tc32.dll';
148: function DIOsetDataEx(h, chip, chan, data:integer):integer; stdcall; external
      'Dio_tc32.dll';
149:
150: function tc08_open_unit (port : integer) : boolean; stdcall; external 'Tc0832';
151:
152: function  tc08_set_channel (port : integer;
153:                             channel : integer;
154:                             tc_type : char;
155:                             factor  : integer;
156:                             offset  : integer;
157:                             slope   : integer) : boolean ; stdcall; external
      'Tc0832';
158: function tc08_get_temp (var value : longint ;
159:                         port : integer;
160:                         channel : integer;
161:                         filtered : boolean) : boolean; stdcall; external
      'Tc0832';
162: procedure tc08_close_unit (port : integer); stdcall; external 'Tc0832';
163:
164:
165: function TimeToInteger(timesent:TDateTime):Longint;
166: var
167: hours, minutes, seconds, milliseconds:word;
168: hours1, minutes1:longint;
169: begin
170: decodetime(TimeSent, hours, minutes, seconds, milliseconds);
171: hours1:=hours;
172: hours1:=hours1*3600;
173: minutes1:=minutes*60;
174: timetointeger:=hours1+minutes1+seconds;
175: end;
176:
177:
178: Function decToBin(decimal:longint):string;
179: var x:longint; binary:string;
180: begin
181: x:=0;
182: if decimal-128>=0 then begin decimal:=decimal-128; x:=x+10000000; end;
183: if decimal-64>=0  then begin decimal:=decimal-64;  x:=x+1000000;end;
184: if decimal-32>=0  then begin decimal:=decimal-32;  x:=x+100000;end;
185: if decimal-16>=0  then begin decimal:=decimal-16;  x:=x+10000;end;
186: if decimal-8>=0   then begin decimal:=decimal-8;   x:=x+1000;end;
187: if decimal-4>=0   then begin decimal:=decimal-4;   x:=x+100;end;
188: if decimal-2>=0   then begin decimal:=decimal-2;   x:=x+10;end;
189: if decimal-1>=0   then begin decimal:=decimal-1;   x:=x+1;end;
190: binary:=intToStr(x);
191: while length(binary)<8 do binary:=concat('0'+binary);
192:  result:=binary;
193: end;
194:
195: function BinTodec(Binary:string):integer;
196: var
197:  deci : integer;
198: begin
199: deci:=0;
200: deci:=StrToInt(Binary[1])*128 ;
201: deci:=deci+(StrToInt(Binary[2])*64);
202: deci:=deci+(StrToInt(Binary[3])*32);
203: deci:=deci+(StrToInt(Binary[4])*16);
204: deci:=deci+(StrToInt(Binary[5])*8);
205: deci:=deci+(StrToInt(Binary[6])*4);
206: deci:=deci+(StrToInt(Binary[7])*2);
207: deci:=deci+(StrToInt(Binary[8])*1);
208: result:=deci;
209: end;
210:
211: function GetDataStringA: string;
212: begin
213: dioGetDataEx(h, 0, 0, @data1);
214: result:=dectoBin(data1);
215:
216:   end;
217:
```

```
218: function GetDataStringB:string;
219: begin
220: dioGetDataEx(h, 0, 1, @data2);
221: result:=dectoBin(data2);
222:  end;
223:
224:  procedure setBit(relay:integer);
225:  var
226:   StatusA, StatusB : String;
227:  begin
228:  StatusA:=GetDataStringA;
229:   StatusB:=GetDataStringB;
230:    if relay<=8 then if StatusA[9-relay]='0' then begin
231:    StatusA[9-relay]:='1'; dioSetDataEx(h, 0, 0, binToDec(StatusA)); end;
232:    if relay>8 then if StatusB[17-relay]='0' then begin
233:    StatusB[17-relay]:='1'; dioSetDataEx(h, 0, 1, binToDec(StatusB)); end;
234: end;//of SetBit function which is User Defined
235:
236: procedure toggleBit(relay:integer);
237:  var
238:   StatusA, StatusB : String;
239:  begin
240:  StatusA:=GetDataStringA;
241:   StatusB:=GetDataStringB;
242:
243:    if relay<=8 then if StatusA[9-relay]='0' then//9 is because last character is
     first relay
244:    StatusA[9-relay]:='1'
245:     else if StatusA[9-relay]='1' then
246:      StatusA[9-relay]:='0';
247:       dioSetDataEx(h, 0, 0, binToDec(StatusA));
248:
249:    if relay>8 then if StatusB[17-relay]='0' then
250:    StatusB[17-relay]:='1'
251:     else if StatusB[17-relay]='1' then
252:      StatusB[17-relay]:='0';
253:       dioSetDataEx(h, 0, 1, binToDec(StatusB));
254: end;//
255:
256: procedure clearBit(relay:integer);
257: var
258:   StatusA, StatusB : String;
259:  begin
260:  StatusA:=GetDataStringA;
261:   StatusB:=GetDataStringB;
262:    if relay<=8 then if StatusA[9-relay]='1' then begin
263:    StatusA[9-relay]:='0'; dioSetDataEx(h, 0, 0, binToDec(StatusA)); end;
264:    if relay>8 then if StatusB[17-relay]='1' then begin
265:    StatusB[17-relay]:='0'; dioSetDataEx(h, 0, 1, binToDec(StatusB)); end;
266: end;//
267:
268:
269:
270:
271:
272:
273: //beginning of main body of program and end of declarations
274:
275: procedure TForm1.Timer1Timer(Sender: TObject);
276: begin
277: globaltime:=timetointeger(time);
278: label4.caption:=timetostr(time);
279:
280: relaystatusStrA:=GetDataStringA;
281: if relaystatusstrA[8]='1' then label5.color:=cllime else label5.color:=clgray;
282: if relaystatusstrA[7]='1' then label6.color:=cllime else label6.color:=clgray;
283: if relaystatusstrA[6]='1' then label7.color:=cllime else label7.color:=clgray;
284: if relaystatusstrA[5]='1' then label8.color:=cllime else label8.color:=clgray;
285: if relaystatusstrA[4]='1' then label9.color:=cllime else label9.color:=clgray;
286: if relaystatusstrA[3]='1' then label10.color:=cllime else label10.color:=clgray;
287: if relaystatusstrA[2]='1' then label11.color:=cllime else label11.color:=clgray;
288: if relaystatusstrA[1]='1' then label12.color:=cllime else label12.color:=clgray;
289: relaystatusStrB:=GetDataStringB;;
290: if relaystatusstrB[8]='1' then label13.color:=cllime else
```

```
     label13.color:=clgray;//Halogen lights
291: if relaystatusstrB[7]='1' then label26.color:=cllime else
     label26.color:=clgray;//sealed fans
292: if relaystatusstrB[6]='1' then label27.color:=cllime else
     label27.color:=clgray;//sealed lights
293: if relaystatusstrB[5]='1' then label36.color:=cllime else
     label36.color:=clgray;//Main Fans
294: if relaystatusstrB[4]='1' then label34.color:=cllime else
     label34.color:=clgray;//Fluorescent lights
295:
296:
297: if TimeCheck<>GlobalTime then begin {prevents code being carried out more than
     once}
298:
299: //begins by examining Sealed Light Unit Times and Switching Appropriately
300:
301: if Globaltime=SealedLightsOn then begin setBit( 11); append(a);
302: WriteLn(a, 'On '+datetostr(date)+' the sealed unit lights came on at
     '+timetostr(time));closefile(a); end;
303:
304: if GlobalTime=SealedlightsOff then begin clearBit( 11); append(a);
305: WriteLn(a, 'On '+datetostr(date)+' the sealed unit lights went off at
     '+timetostr(time));closefile(a); end;
306:
307: //Compares Sealed Unit, Ambient and reference Temperature and switches Fans
     accordingly
308:
309: if StrToFloat(EditCutOutTemp.text)>Temp1 then
310: if FansOn=true then begin clearBit( 10);  FansOn:=false; end;
311:
312: if StrToFloat(EditCutOutTemp.text)<=Temp1 then
313: if FansOn=False then begin  setBit( 10); FansOn:=true;  end;
314:
315:  if StrToFloat(EditCutOutTemp.text)>Temp2 then
316: if RoomFansOn=true then begin clearBit( 12); RoomFansOn:=false; end;
317:
318: if StrToFloat(EditCutOutTemp.text)<=Temp2 then
319: if RoomFansOn=False then begin  setBit( 12); RoomFansOn:=true; end;
320:
321: if RG1.itemindex=0 then begin {Sigmoidal Routine}
322:
323: if GlobalTime=BeginDawn then begin setBit( 1); append(a);
324: WriteLn(a, 'On '+datetostr(date)+' the first little light came on at
     '+timetostr(Time));closefile(a);end;
325:
326: if GlobalTime=beginDawn+lightInterval then begin setBit( 8);append(a);
327: WriteLn(a, 'On '+datetostr(date)+' the second little light came on at
     '+timetostr(Time));closefile(a);end;
328:
329: if GlobalTime=BeginDawn+(LightInterval*2) then begin setBit( 4); append(a);
330: WriteLn(a, 'On '+datetostr(date)+' the third little light came on at
     '+timetostr(Time));closefile(a);end;
331:
332: if GlobalTime=BeginDawn+(LightInterval*3) then begin setBit( 5); append(a);
333: WriteLn(a, 'On '+datetostr(date)+' the fourth little light came on at
     '+timetostr(Time));closefile(a);end;
334:
335: if GlobalTime=BeginDawn+(LightInterval*4) then begin setBit( 2); append(a);
336: WriteLn(a, 'On '+datetostr(date)+' the fifth little light came on at
     '+timetostr(Time));closefile(a);end;
337:
338: if GlobalTime=BeginDawn+(LightInterval*5) then begin setBit( 7);append(a);
339: WriteLn(a, 'On '+datetostr(date)+' the sixth little light came on at
     '+timetostr(Time));closefile(a);end;
340:
341: if GlobalTime=BeginDawn+(LightInterval*6) then begin setBit( 3); append(a);
342: WriteLn(a, 'On '+datetostr(date)+' the seventh little light came on at
     '+timetostr(Time));closefile(a);end;
343:
344: if GlobalTime=BeginDawn+(LightInterval*7) then begin setBit( 6); append(a);
345: WriteLn(a, 'On '+datetostr(date)+' the eigtht little light came on at
     '+timetostr(Time));closefile(a);end;
346:
347: if GlobalTime=BeginDawn+(LightInterval*8) then begin setBit(13); append(a);
```

```
348: WriteLn(a, 'On '+datetostr(date)+' the fluorescent lights came on at
     '+timetostr(Time));
349: WriteLn(a, '    ');{writes a blank line} closefile(a);end;
350:
351: if GlobalTime=BeginDawn+(LightInterval*9) then begin setBit(9); append(a);
352: WriteLn(a, 'On '+datetostr(date)+' the halogen floodlights came on at
     '+timetostr(Time));
353: WriteLn(a, '    ');{writes a blank line} closefile(a);end;
354:
355:  {*************Dusk events**************}
356: if GlobalTime=BeginDusk then begin clearBit(9); append(a);
357: WriteLn(a, 'On '+datetostr(date)+' the halogen floodlights went off at
     '+timetostr(Time));
358: closefile(a);end;
359:
360: if GlobalTime=BeginDusk+(LightInterval) then begin clearBit(13); append(a);
361: WriteLn(a, 'On '+datetostr(date)+' the big bright lights went off at
     '+timetostr(Time));
362: closefile(a);end;
363:
364: if GlobalTime=BeginDusk+(LightInterval*2) then begin clearBit( 6); append(a);
365: WriteLn(a, 'On '+datetostr(date)+' the first little light went off at
     '+timetostr(Time));
366: closefile(a); end;
367:
368: if GlobalTime=BeginDusk+(LightInterval*3) then begin clearBit( 3); append(a);
369: WriteLn(a, 'On '+datetostr(date)+' the second little light went off at
     '+timetostr(Time));
370: closefile(a); end;
371:
372: if GlobalTime=BeginDusk+(LightInterval*4) then begin clearBit( 7); append(a);
373: WriteLn(a, 'On '+datetostr(date)+' the third little light went off at
     '+timetostr(Time));
374: closefile(a); end;
375:
376: if GlobalTime=BeginDusk+(LightInterval*5) then begin clearBit( 2); append(a);
377: WriteLn(a, 'On '+datetostr(date)+' the fourth little light went off at
     '+timetostr(Time));
378: closefile(a); end;
379:
380: if GlobalTime=BeginDusk+(LightInterval*6) then begin clearBit( 5); append(a);
381: WriteLn(a, 'On '+datetostr(date)+' the fifth little light went off at
     '+timetostr(Time));
382: closefile(a); end;
383:
384: if GlobalTime=BeginDusk+(LightInterval*7) then begin clearBit( 4); append(a);
385: WriteLn(a, 'On '+datetostr(date)+' the sixth little light went off at
     '+timetostr(Time));
386: closefile(a); end;
387:
388: if GlobalTime=BeginDusk+(LightInterval*8) then begin clearBit( 8); append(a);
389: WriteLn(a, 'On '+datetostr(date)+' the seventh little light went off at
     '+timetostr(Time));
390: closefile(a); end;
391:
392: if GlobalTime=BeginDusk+(LightInterval*9) then begin clearBit( 1); append(a);
393: WriteLn(a, 'On '+datetostr(date)+' the last little light went off at
     '+timetostr(Time));
394: WriteLn(a, '    '); closefile(a); end;
395:
396: end{of 'if sigmoidal' condition}
397:
398: else if RG1.itemindex=1 {i.e. rectangular} then begin
399:
400: if Globaltime=begindawn then begin dioSetDataEx(h, 0, 0, 255);
401: SetBit(9); SetBit(13); //switches first 9 relays and relay 13 =decimal 17 or
     binary 10001
402: append(a); writeln(a, 'Rectangular: On '+datetostr(date)+' all the lights came
     on at '+timetostr(time));
403: closefile(a); end;
404:
405: if GlobalTime=RecDusk then begin DiosetDataEx(h, 0, 0, 0);
406: ClearBit(9); ClearBit(13); append(a);//Must Switch 1 to 8, 9 & 13 but nothing
     else
```

```
407: writeLn(a, 'Rectangular: On '+datetostr(date)+' all the lights went off at
     '+timetostr(time));
408: closefile(a); end;
409:
410: end; {end of 'if rectangular' condition};
411: end;
412: Timecheck:=GlobalTime;
413: end;
414:
415:
416: procedure TForm1.Exit1Click(Sender: TObject);
417: begin
418: if messagedlg('Is this what you want, what you really, really want?',
419: mtinformation, [mbYes, mbNo], 0) =mryes then begin
420:     tc08_close_unit (comport);
421:     opened := FALSE;
422:
423: close; end;
424: end;
425:
426: procedure TForm1.ReselectParameters1Click(Sender: TObject);
427: begin
428: edit1.enabled:=true;
429: edit2.enabled:=true;
430: edit3.enabled:=true;
431: edit4.enabled:=true;
432: edit5.enabled:=true;
433: editCutOutTemp.enabled:=true;
434: button1.enabled:=true;
435: BtnSetFansEtc.enabled:=true;
436: end;
437:
438: procedure TForm1.Button1Click(Sender: TObject);
439: begin
440: edit1.enabled:=false;
441: edit2.enabled:=false;
442: edit3.enabled:=false;
443: button1.enabled:=false;
444: dawn:=strtotime(edit1.text);
445: begindawn:=timetointeger(dawn);{time of dawn in seconds}
446: daylength:=3600*strtoint(edit2.text); {daylength in seconds}
447: DawnDuskLength:=60*strtoint(edit3.text); {dawn/dusk length in seconds}
448: lightinterval:=dawndusklength div 9; //there are now 10 switched lights total
449: begindusk:=begindawn+daylength-dawndusklength;
450: recdusk:=begindawn+daylength;
451: end;
452:
453: procedure TForm1.About1Click(Sender: TObject);
454: begin
455: aboutbox.show;
456: end;
457:
458:
459: procedure TForm1.ClearLogFile1Click(Sender: TObject);
460: begin
461: if messagedlg('Now are you sure about this?',
462: mtinformation, [mbYes, mbNo], 0) =mryes then
463: rewrite(a);
464: end;
465:
466: procedure TForm1.ToggleAllClick(Sender: TObject);
467: var i:integer;
468: begin
469: for i:=1 to 8 do togglebit(i);
470: end;
471:
472: procedure TForm1.Button3Click(Sender: TObject);
473: begin
474: togglebit(1);
475: end;
476:
477: procedure TForm1.Button4Click(Sender: TObject);
478: begin
479: togglebit(2);
```

```
480: end;
481:
482: procedure TForm1.Button5Click(Sender: TObject);
483: begin
484: togglebit(3);
485: end;
486:
487: procedure TForm1.Button6Click(Sender: TObject);
488: begin
489: togglebit(4);
490: end;
491:
492: procedure TForm1.Button7Click(Sender: TObject);
493: begin
494: togglebit(5);
495: end;
496:
497: procedure TForm1.Button8Click(Sender: TObject);
498: begin
499: togglebit(6);
500: end;
501:
502: procedure TForm1.Button9Click(Sender: TObject);
503: begin
504: togglebit(7);
505: end;
506:
507: procedure TForm1.Button10Click(Sender: TObject);
508: begin
509: togglebit(8);
510: end;
511:
512: procedure TForm1.Button11Click(Sender: TObject);
513: begin
514: togglebit(9);
515: end;
516:
517: procedure TForm1.SHClick(Sender: TObject);
518: begin
519: if SH.Checked=false then begin
520: SH.checked:=true;
521: Form1.ShowHint:=true;
522: end
523: else if SH.Checked=true then begin
524: SH.checked:=False;
525: Form1.ShowHint:=False;
526: end;
527: end;
528:
529: procedure TForm1.BtnSetFansEtcClick(Sender: TObject);
530: begin
531: BtnSetFansEtc.enabled:=false;
532: Edit4.enabled:=false;
533: Edit5.enabled:=false;
534: EditCutOutTemp.enabled:=false;
535: SealedLightDawn:=strtoTime(edit4.text);
536: SealedLightDusk:=strToTime(edit5.text);
537: SealedLightsOn:=TimetoInteger(SealedLightDawn);
538: SealedLightsOff:=TimeToInteger(SealedLightDusk);
539: end;
540:
541: procedure TForm1.TempTimerTimer(Sender: TObject);
542: begin
543: if opened then
544:   begin
545:     ok := tc08_get_temp (temp, comport, 1, FALSE);
546:     if ok then
547:       begin
548:         ftemp := temp / 100.0;
549:         str (ftemp: 6:2, s);
550:          EditChan1.text := s;
551:          temp1:=ftemp;
552:       end
553:     else
```

```
554:       EditChan1.text := '';
555:
556:     ok := tc08_get_temp (temp, comport, 2, FALSE);
557:     if ok then
558:       begin
559:         ftemp := temp / 100.0;
560:         str (ftemp:6:2, s);
561:         EditChan2.text := s;
562:         temp2:=ftemp;
563:       end
564:     else
565:       EditChan2.text := '';
566:         end;
567: if EditChan2.text<>'' then
568:   begin//i.e. if Data is being displayed
569: Append(TempFile);
570:  WriteLn(TempFile, DateToStr(date)+', '+ TimeToStr(Time)+', '+EditChan2.text+',
      Temperature in Sealed Unit is '+EditChan1.text);
571:   CloseFile(TempFile);         end;
572:
573: end;
574:
575:
576:
577: procedure TForm1.Button2Click(Sender: TObject);
578: begin
579: togglebit(10);
580: end;
581:
582: procedure TForm1.Button12Click(Sender: TObject);
583: begin
584: togglebit(11);
585: end;
586:
587: procedure TForm1.EditTempChange(Sender: TObject);
588: begin
589: temptimer.interval:=(StrToInt(editTemp.text)*1000);
590: end;
591:
592: procedure TForm1.Button13Click(Sender: TObject);
593: begin
594: togglebit(13);//fluorescent lights
595: end;
596:
597: procedure TForm1.Button14Click(Sender: TObject);
598: begin
599: togglebit(12);
600: end;
601:
602: end.
```